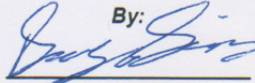


Kairos Odyssey

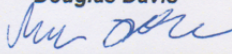
Interactive Media and Game Development

**A Major Qualifying Project Report
submitted to the faculty of
WORCESTER POLYTECHNIC INSTITUTE**

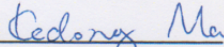
By:



Douglas Davis



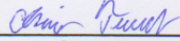
Mark Foster



Kedong Ma



Andrew Strout

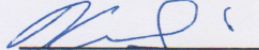


Chris Turner

Advised By:



Professor Ralph Sutter



Professor Keith Zizza

Abstract

Kairos Odyssey is a linear, first-person puzzle adventure game set in a landscape frozen in time. Through the use of the Chronosphere and Tether, the player must solve time and space-altering puzzles on their journey to repair the World Clock and restore time's flow. For our MQP, we set out to create a vertical slice of *Kairos Odyssey* in the Unreal Engine 4 game development kit to demonstrate the game's mechanics and visuals.

Acknowledgements

Our advisors, Professor Keith Zizza and Professor Ralph Sutter, were essential to this project. We'd like to thank them for their guidance and support.

Table of Contents

- Abstract
- Acknowledgements
- Table of Contents
- Table of Figures
- 1 - Introduction
- 2 - Design
 - 2.1 - Setting and Story
 - 2.1.1 - Background
 - 2.1.2 - Character
 - 2.1.3 - Environment
 - 2.2 - Artistic
 - 2.2.1 - Architecture
 - 2.2.2 - Gauntlet
 - 2.3 - Gameplay
 - 2.3.1 - Mechanics
 - 2.3.2 - Puzzles
 - 2.3.3 - Controls
 - 2.4 Music and Sound
 - 2.4.1 - Music
 - 2.4.2 - Sound
 - 2.5 - User Interface
- 3 - Implementation
 - 3.1 - Scope
 - 3.2 - Tools
 - 3.2.1 - Engine
 - 3.2.2 - Source Control
 - 3.3 - Workflow
 - 3.3.1 - Asset Creation
 - 3.3.2 - Animation
 - 3.3.3 - Gameplay Development
 - 3.4 - Integration
 - 3.4.1 - Animation
 - 3.4.2 - Level
- 4 - Playtesting
 - 4.1 - PAX
- 5 - Post-Mortem
 - 5.1 - Doug Davis
 - 5.2 - Mark Foster
 - 5.3 - Kedong Ma
 - 5.4 - Andrew Strout

5.5 - Chris Turner:

6 - Conclusion

6 - Appendices

6.1 - History of the World Clock

6.2 - Original Pitch Document

6.3 - Notebook Samples

Table of Figures

Figure 1: World Clock concept.	8
Figure 2: Character concept art. Left: Male, Right: Female.	9
Figure 3: Left: Female character facial detail. Right: Example of Albanian traditional clothing.	10
Figure 4: Environmental concept art.	11
Figure 5: Example of Romantic depiction of Gothic architecture.	12
Figure 6: Modular environment asset list.	13
Figure 7: Gauntlet Model.	14
Figure 8: Chronosphere art. Left: concept art, Right: in-game appearance.	16
Figure 9: The Tether's final in-game appearance.	17
Figure 10: The large piece of the Archway frozen before it could fall.	19
Figure 11: The Ramp of the Ball Roll puzzle with the ball up top.	20
Figure 12: The Tether already attached to the shrine, awaiting the Chronosphere.	21
Figure 13: Part 1 of the Final Puzzle	22
Figure 14: The second part of the final puzzle.	23
Figure 15: All elements of the UI displayed on the player's character.	28
Figure 16: A screenshot of our finished demo in the Unreal Engine 4 Editor.	31
Figure 17: The completed character hand model in 3DS Max.	34
Figure 18: The bone structure of the player arm.	35
Figure 19: The Blueprint scripting for the Chronosphere.	37
Figure 20: The Chronosphere prototype.	38
Figure 21: The Tether prototype.	39
Figure 22: A demonstration of the cart-pushing mechanics.	41
Figure 23: Our rope prototype, as seen in our PAX application video.	42

1 - Introduction

Kairos Odyssey is a 3D, first-person, puzzle-platformer set in a world frozen in time. The player must navigate through a variety of time based puzzles using abilities that allow them to control time and space. With these abilities the player may manipulate the environment and progress through the game. The concept for *Kairos Odyssey* was created in April, 2014 and was developed using the Unreal Engine 4 game development kit.

The team for *Kairos Odyssey* was created by Andrew Strout and Douglas Davis, who had a vision to create a first-person puzzle game as their MQP. Andrew would take the role of lead programmer and Douglas was the lead artist and environment designer. To round out the group, Andrew invited Mark Foster and Chris Turner to join him on the tech team for their programming knowledge and insights on puzzle and level design. Doug invited Kedong Ma to join him on the art team as a character artist, animator and concept artist.

When we began brainstorming in April, we first started with a story. We wanted our 'seed' for the game to come from a compelling yet simple narrative that allowed us to explore an interesting game mechanic. Initially, we were very inspired by the elaborate Greek mythology surrounding the titan Cronus, as well as the more modern depictions of Father Time, the personification of time itself. We were drawn to the dark and often absurdist nature of these stories and felt that these could inspire a unique setting and aesthetic for our game. We also felt that the theme of time in both stories could provide an interesting backbone to the gameplay.

After our first few meetings in April we agreed that time would be a central mechanic in our game. We chose time because it presented a variety of unique gameplay challenges that could be interesting and engaging for the player to solve. Specifically, we were attracted to the idea of solving puzzles in a space where time couldn't move forward, and the challenge that this presented to our tech team. What tools would the player need to traverse this environment? And what challenges could be presented to the player to subvert their expectations and urge them to think creatively to solve these puzzles? In addition, our art team was inspired by the aesthetic of a world frozen in time. What visual and auditory cues

could be created to make the player feel like the world around them was paused in time? These questions drove us forward through much of the early production of *Kairos Odyssey*.

After settling on time as the core mechanic, we began to narrow down the exact gameplay and narrative of our game. One of our first core decisions was to make the game a first-person experience. Though various games have used time as a central mechanic in gameplay, we felt we could offer something unique to the player by placing a strong emphasis on immersion in the game world and by giving the player abilities that were visceral, fun to use, and visually arresting. The experience of the player became a strong focus for us, and having the gameplay take place in first-person allowed us the most control over that experience.

To decide the individual abilities the player had, we specifically avoided giving the player too many options. With the inclusion of time-based gameplay, we knew we wanted to avoid overcomplicating the puzzles with too many abilities or mechanics. We looked at and discussed the abilities the player would need in order to solve puzzles in the game world. After a couple weeks of deliberation, we boiled these down to two things: the ability to allow time to move forward and the ability to move an object through space. During this phase we began to adopt a core design philosophy of elegant simplicity. A simpler design means that the game would be easier to understand by players who may possibly approach the game with no prior knowledge. The complexity of the game would emerge from these simple mechanics as they are applied to different situations. This emergent gameplay, in which complexity arose from simple mechanics, was what we deemed elegant simplicity.

From the beginning, we knew that it would be impossible to create the entire game that we were envisioning within the timespan of the MQP itself. The full game would still be planned out to some degree, but we knew that for the sake of our project we had to scope down our game into a smaller demo level that demonstrated the experience of playing the full game. We realized that it would be a good idea to create a vertical slice of our imagined full game; something representative of all facets of our project.

After deciding upon a first-person, puzzle and platforming game focused around time mechanics, we decided upon the final aspects of the narrative. Late in April we began revising

our initial concept for the story, aiming to create a simpler, but equally as compelling, narrative to accompany the players journey through the world. By the end of April we had crafted the basic concept for the game *Kairos Odyssey*. We spent that summer continuing to refine our ideas for the project and began development of the game in August of that year.

2 - Design

2.1 - Setting and Story

2.1.1 - Background

In *Kairos Odyssey*, the player controls an unnamed protagonist who has been tasked by the Gods with repairing the World Clock, a massive monument built by the Gods and placed in the world so that mortal men could perceive time and space. When the structure was placed, mankind tried to harness the monument's power, believing they could understand its energy. In doing so, they broke the tower, causing time itself to cease moving forward. Now the player has been chosen to journey to the heart of the monument and repair the structure from within.

concept.



Figure 1: World Clock

2.1.2 - Character

The protagonist of *Kairos Odyssey* is a hero chosen by the Gods to restore time to the world. His backstory is never fully explained, and his name and identity remain unknown. Though our project only gives a brief vertical slice of the experience, in our hypothetical complete game the player would be shown a brief introductory cutscene. In this cutscene, they would witness the World Clock being broken, and watch as the protagonist's younger sibling, and the world around them, became frozen in time, giving the player a simple yet powerful image to motivate them through the game. Other than then this initial glimpse, the protagonist remains a blank slate. By doing this we remove any narrative baggage that he would otherwise come with, allowing the player to fill in his backstory with their own imagination, and keeping the narrative focused on repairing the World Clock. The intro

cutscene to the game would finish with the player traveling towards the foot of the World Clock and arriving at the entrance to a series of underground caves.

Although the game is played in first-person and the protagonist's right arm is the only part of their body which can be seen by the player, we created a complete design concept of the main character for the sake of reference and for future development goals, such as for use in a third-person cutscene. We also wanted to make sure that we, as the developers, had a complete idea of the protagonist that was cohesive with the game world and narrative.



Figure 2: Character concept art. Left: Male, Right: Female.

Kairos Odyssey is set in a fantasy world without a defined time and location. Our character design started from a female character, inspired by characters from mythologies of different cultures such as the warrior maiden Camilla from Roman mythology, and Mulan from Chinese mythology. The costumes are heavily inspired by traditional Albanian costumes which are highly decorated with ornate, golden details, hanging beads, and cultural motifs. The ornate costumes imply a sense of spiritual honor or purpose, which aligned perfectly with our narrative of a chosen hero. We also liked this because of its unfamiliarity and exotic nature. Visually, it feels rooted in a large amount of history, culture and tradition.



Figure 3: Left: Female character facial detail. Right: Example of Albanian traditional clothing.

Early in development, however, we decided upon using a male character instead of a female character for our demo. The design is mostly based on the same principle as the female character. Inspired by games like *Enslaved: Odyssey to the West*, the character is athletic and half-clothed with shoulder armor on his right side. The entire design has minimal armor pieces to avoid the character seeming like a warrior or fighter.

2.1.3 - Environment

When designing the environments of *Kairos Odyssey*, there were several things that we had to take into consideration. The world of *Kairos Odyssey* is not intended to be Earth, but rather an alternative universe with its own mythos and history. We wanted to avoid creating a world founded on the various tropes of typical fantasy settings, and instead rooted our world in the rich fictional elements found in ancient stories from various cultures. Specifically, we looked at the various worlds depicted in Greek mythology and ancient Chinese legends and fables. We looked to these stories because rather than telling narratives based around realism, they place higher importance on allegory and poetic symbolism; the

relationship between man and the Gods and the origin of the two. These fables helped us establish a world that operated on elemental forces rather than logic, which fit perfectly with our narrative.

After deciding upon the aesthetic of our world, we needed to create a look and feel for the environments that fit the concept; a landscape that felt ancient, unfamiliar and desolate but was rooted in something real and organic. Once again we wanted to carefully straddle the line between realism and fantasy. It was important that the player be in awe of their surroundings; to feel as though they were a miniscule part of the world and remind them of their role as the chosen hero who must complete this daunting task. For inspiration, we looked to a variety of environments across the world. In particular, we drew inspiration from three main locations: the Hebridean coastlines of Ireland, the vast caverns and cave systems found in the mountains of Malaysia, and the beautifully primitive landscapes of Iceland. The topography of our game assets and environments were directly inspired by these locations.



Figure 4: Environmental concept art.

2.2 - Artistic

2.2.1 - Architecture

Though *Kairos Odyssey* takes the player through a variety of environments, including organic underground cave systems, it was important that every level have some type of architecture built into the environment. This was done to remind the player that the entire journey towards the World Clock had been crafted by the Gods to test their abilities. When deciding upon the specific look and feel for the architecture in the game we looked for an aesthetic that blended seamlessly into the landscape; as if it had been there almost as long as the rocks themselves.

For this we began to look at Romantic era depictions of Gothic architecture. Gothic, a style that emerged in the late medieval era, is notable for its pointed arches and grand structures with ornate detailing. Stylistically, the tall heavenward arches and spires of Gothic architecture suggest a sort of reverent tribute to the heavens above. However, these characteristics are further exaggerated in the paintings of artists from the Romantic era. Their depictions of Gothic architecture simultaneously exaggerate the grandeur and visual drama of the structures, as well as their fragility and susceptibility to the forces of time itself.¹ The combination was a perfect fit for the aesthetic of our architecture.



Figure 5: Example of Romantic depiction of Gothic architecture.²

¹ "The Seven Key Characteristics of Gothic Architecture."
http://www.exploring-castles.com/characteristics_of_gothic_architecture.html

² Friedrich, David Caspar. "Cloister Cemetery in the Snow."

Because we would be using the architectural assets throughout the entirety of the demo, we decided to create a large set of modular building assets that could be combined in various ways. There are a wide range of modular architectural pieces in *Kairos Odyssey*. Each model uses a variation of an aged stone texture, depending on where in the level it is, and a color theme that corresponds with the cave environment. To keep our assets organized, we created excel sheets that documented the state of most our assets, as seen in Figure 6 below.



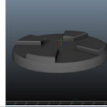



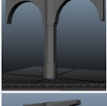


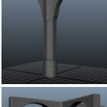


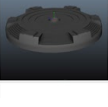

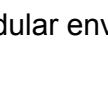
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
		Standard arch	Architectures	Arch		X	X				Arch connector quad (1x4)	Architectures	Arch Connector Quad		X			Platform top (back) of quad arch support	Ar	
2		Standard arch quad	Architectures	Arch			X	X			Platform base	Architectures	Platform		X			Pillar	Ar	
3		Arch connector (left)	Architectures	Arch Connector			X				Platform top (back)	Architectures	Platform		X			Pillar	Ar	
4		Arch connector (right)	Architectures	Arch Connector			X				Platform top (back)	Architectures	Platform		X					
5		Arch connector (quad)	Architectures	Arch Connector			X				Platform top (back)	Architectures	Platform		X					
6		Arch connector quad (left)	Architectures	Arch Connector Quad			X				Platform top (left) of quad	Architectures	Platform		X					

Figure 6: Modular environment asset list.

2.2.2 - Gauntlet

To create an immersive game play experience, our team chose to remove many heads-up display elements and integrate the user interface with the character model itself. Taking inspiration from games like *Portal* and *Mirror's Edge*, the player's gauntlet is designed to have all the visual representation of in-game elements and information.

Based on the history of World Clock, the design of the character reflects his duty as a savior of the world. The design of the gauntlet is very symmetrical and the material we chose is metal, which relates to strength and durability. The decoration design on the gauntlet was influenced by medieval armor. Initially we created several designs of the glove, with varying degrees of armor. We decided to go with the current design as it was more elegant and wouldn't complicate the UI aspect that the glove needed to accomplish. The gauntlet is

divided into three separate parts: the Chronosphere bracer, the Tether connector and wrist armor.



Figure 7: Gauntlet Model.

The Chronosphere bracer houses the Chronosphere and a small clock. The clock design was inspired by Chinese traditional craftwork of gold and jade jewelry. We decided on the theme of having the Chronosphere rest on the wrist like a watch. This went well with the idea of the player literally having time on their wrist. The Chronosphere bracer is the core of the glove; designed to draw the player's attention, it has the most complicated design compared to other parts.

The Tether connector is another important device of the gauntlet, as it houses the Tether. The Tether connector is located on the back of the wrist armor running along the forearm. This section was designed to be open to provide space for the Tether to later occupy, while also accenting its importance. The rear part of the gauntlet was also heavily decorated with a medieval European pattern to be visually consistent with the Gothic style of

the architecture. Because the front already had the Chronosphere, the back needed to be equally embellished to keep the glove design balanced.

2.3 - Gameplay

2.3.1 - Mechanics

When we first began brainstorming for our game, we knew we wanted mechanics that dealt with time manipulation. One early idea was that the player would stay in one small area but travel back and forth between the days of the week, exploring the results of making different decisions at earlier points in the week. Because of its complexity, the idea was scrapped in favor of gameplay that involved directly manipulating time through physics. We soon had the idea of the Chronosphere: the game world is frozen in time, but the player can place a sphere in the world in which time passes as normal. This allows the player to use objects in the world as both static and dynamic means to solve puzzles. This idea quickly became the primary focus of our game.

An important characteristic of freezing and unfreezing an object in time is the conservation of momentum. If a ball is moving through the air and then becomes frozen in time, it will halt its motion and float in midair. Once time resumes the ball will continue moving at the same velocity it had right as time froze, following its same flight path. This is an important distinction that ensures the Chronosphere is a mechanic that clearly affects an object in time, and doesn't just affect its motion.

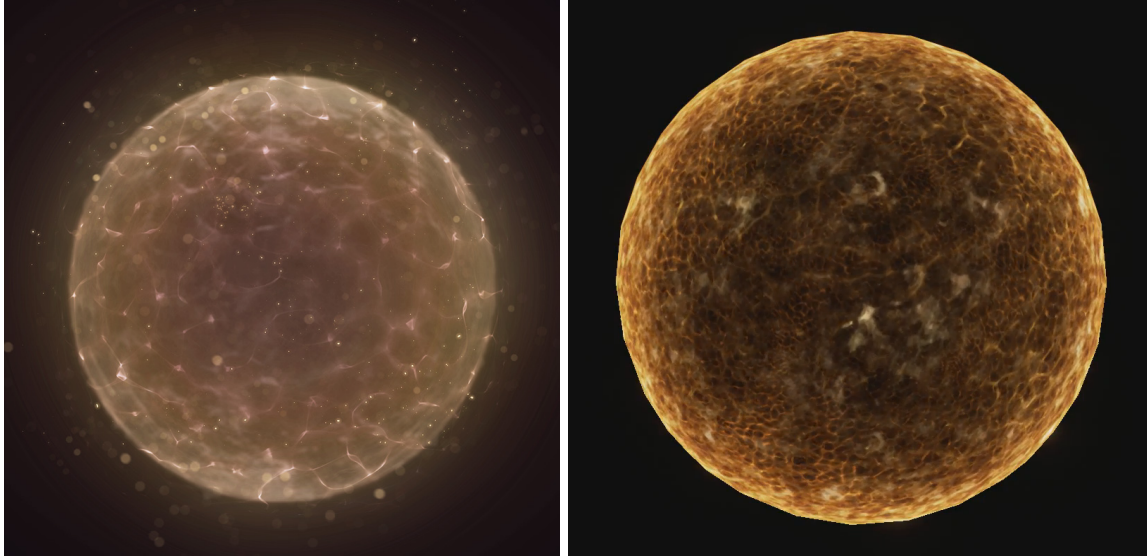


Figure 8: Chronosphere art. Left: concept art, Right: in-game appearance.

As we discussed different ways to use the Chronosphere, we came to a realization. While we still really liked the Chronosphere idea, we determined that it couldn't carry a puzzle game by itself. What the Chronosphere essentially does is allow an object to apply its potential energy. For instance, objects frozen in time in the air have potential energy due to gravity. Once unfrozen, they apply that energy by falling to the ground. We realized that if the Chronosphere was our game's only mechanic, all the player could do was make objects fall down. Gravity was the only way objects were going to have any potential energy. The only way anything interesting was going to happen was if things were already set up to interact with each other in a meaningful way and only needed some sort of "push" to get started. While these Rube-Goldberg-machine puzzles could be visually interesting, they were hardly interactive. The puzzle solution would already be set up, all the player would have to do is get it going.

So consequently, we began to brainstorm ideas to give the player some way to manipulate objects along with the Chronosphere. Up until this point, all of the puzzle solutions would have to be set up in the level already, but we wanted to give the player a means to perform that set-up. That would result in challenging, meaningful puzzle-solving. We found our solution in the form of the Tether, a beam that the player creates between two objects. When the Tether is attached to two objects, they are pulled toward one another. However, the objects won't be able to move at all unless they are within the boundaries of the

Chronosphere. Not only does this ability open up many possibilities for the player to manipulate objects, it also synergizes perfectly with the Chronosphere. However, the Tether has a limited attachment distance, so objects must be within a certain range to be connected. Additionally, the Tether can only be connected to movable objects and special “hooks” placed in the world, and cannot be linked to the floor, walls, or other structural pieces. This prevents the player from just taking a boring and tedious approach to every puzzle while still providing a variety of choices when approaching a challenge.

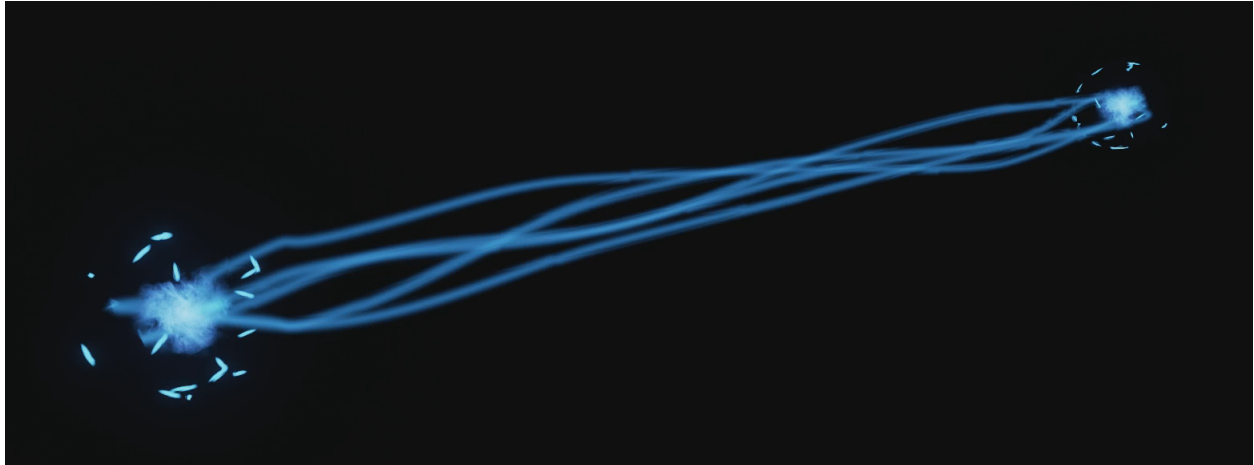


Figure 9: The Tether's final in-game appearance.

Consider a hypothetical room where the goal is to move an object to a certain location. With walls and ceilings being tetherable, the player could use the Tether to drag the object to the ceiling, move it above the correct location, and drop it in place, ignoring any obstacles. This degenerate solution would work no matter what obstacles are in this room, unless it is specifically prevented by the layout of the room. It would be much harder to design any rooms if every aspect of the puzzle design had such a large impact on the environment design.

One of our abandoned ideas was to give the player the ability to toggle an object's collision properties, allowing it to pass through other objects. This was abandoned after concerns were raised regarding unpredictable game behavior. If we couldn't even figure out how it would work, the player certainly wouldn't be able to. It was difficult to determine exactly how this mechanic would interact with any of the other ones, which eventually led to us scrapping this idea. After thinking about it more, we decided that the Chronosphere and Tether together were enough to make a substantial puzzle game.

The gameplay of *Kairos Odyssey* is focused on solving puzzles using the Chronosphere and Tether mechanics available to the player. Objects without the Chronosphere placed on them remain motionless, allowing the player to carefully examine the situation before attempting to change it. By combining use of the Tether and the Chronosphere, the player can manipulate their environment, solve puzzles, and move forward in the game.

2.3.2 - Puzzles

Puzzles were always a central focus for the gameplay of *Kairos Odyssey*, and designing them began very early in the development of the game. With each new power that was planned and designed, our team had to imagine different puzzles to utilize those powers. Our three programmers were responsible for creating those puzzles and testing them out in the early sandbox level we designed.

While designing those early puzzles, we often found unforeseen ways to solve those puzzles while playtesting. This was fine in some cases, but this also shed light on some balancing issues with the strength of one power in particular; the Tether. Designing puzzles with the knowledge that a player could just pull an object to any part of a level became too difficult. Trying to design puzzles in a 3D space where the walls, floor, and ceiling directly worked against the puzzle caused us to reevaluate that power. It was through this design process of building and testing puzzles that caused us to change the Tether to only attach to objects and anchor points. This opened up the possible puzzles our team could make. We now could create puzzles with more control over how the player could solve them.

In our level, we use puzzles to teach our players the mechanics of the game they can control and later test their mastery of those mechanics. There are three main puzzles in our level that can be broken down into two types of puzzles: Player-Based puzzles and Object-Based puzzles.

The first puzzle is the bridge puzzle that teaches the player about the time-flow properties and the attaching properties of the Chronosphere. After the player receives the Chronosphere, they walk down a hallway and are presented with a broken bridge that has a gap too large to jump over. At this point the player has already been instructed how to use the Chronosphere, but has yet to use it on any objects in the level. In front of the bridge is an archway that has been shattered and frozen in time before the pieces could fall. When the player looks at these pieces they will first notice a blue outline around the pieces of the archway that indicate that the object can be unfrozen via the Chronosphere and later attached via the Tether. The player will then fire the Chronosphere at one of the three archway pieces and will cause them to resume their momentum; causing them to fall. The largest piece will fall and form a bridge over the gap that the player can jump up on and cross. This puzzle demonstrates that objects frozen in time conserve their momentum, that interactable objects that can be unfrozen have a blue outline when looked at, and finally that the Chronosphere can be attached to objects and travel with them.



Figure 10: The large piece of the Archway frozen before it could fall.

Our second puzzle was the Ball Roll puzzle which further demonstrates the lessons taught by the previous puzzle. In this puzzle, the player will need to attach the Chronosphere to the Puzzle Ball atop a slide. Once they do so, they must then push the Puzzle Ball down the slide, where it will then roll into a round socket. This will finish the puzzle and demonstrate the Lock and Key aspect of the puzzle, where moving the Puzzle Ball into the proper place will unlock a door and allow for further progression through the level.



Figure 11: The Ramp of the Ball Roll puzzle with the ball up top.

The third puzzle in our level has an elegance that our team is quite proud of. The player approaches a bridge with another large gap that has shrine floating between the two sides. Attached to the shrine and an anchor on the same side as the player is the Tether. Here the player is first introduced to the power they will be receiving and their only action is to fire the Chronosphere at the shrine. Upon doing so, the Shrine will become unfrozen and the Tether will pull it towards the player, thus demonstrating the Tether's ability to pull objects towards anchors. Once the player moves onto the shrine, they will receive the Tether and be informed how to use it. They will then have to use the Tether to connect the Shrine they are standing on and the anchor on the other side of the bridge in order to cross. This puzzle

teaches the player about how the Tether pulls objects towards anchors and that objects cannot be pulled while not within the Chronosphere, but it manages to display all this information before the player even receives the new power. After seeing this demonstration, the player gains the power and basically has to recreate the demonstration to get to the other side of the gap, showing they understand how the ability works.



Figure 12: The Tether already attached to the shrine, awaiting the Chronosphere.

The fourth and final puzzle of our level was a two part puzzle where the player had to utilize the momentum saving properties of the Chronosphere and the pulling aspect of the Tether to move a Puzzle Ball from the bottom of the puzzle to the top, in order to unlock the final door. The puzzle started out with a small cliff like structure with a round base that drained to a semi-cylindrical cut-out on the cliff's face. At the bottom and in the middle of the cylinder cut-out were two Tether anchors. At the top of the cliff was a round slot that the Puzzle Ball could rest in with a Tether anchor above it. The Puzzle Ball was initially placed in front of the round base and upon applying the Chronosphere would roll down the base to the bottom of

the cylindrical cut-out. From here the player had to get the Puzzle Ball to the top of the cliff and into the resting slot. The player could do this by attaching the Chronosphere to the Puzzle Ball and using the Tether to directly connect the Puzzle Ball to the anchor directly above the resting slot, or they could attach the Puzzle Ball to the middle anchor and then break the Tether to have the Puzzle Ball continue its leftover momentum from the Tether to carry it above the cliff. The more likely solution the player would have used here was the former, as this was the first situation where they had to combine the two powers.



Figure 13: Part 1 of the Final Puzzle.

Once at the top of the first cliff, the player has reached the second part of the puzzle. Here the player would find another cliff similar to the first, only much larger and with a box cut-out toward the top of the cylindrical cut-out. The player would then use the Tether to get the Puzzle Ball out of the resting slot and into the round base of the second part of the puzzle, and would then begin trying to solve it. If they attempted to attach the Puzzle Ball to the anchor directly above the resting slot atop the second cliff, the Puzzle Ball would be pulled up the cylindrical cut-out like the previous cliff but would become stuck in the box cut-out at the

top. Here the player would realize that a strategy that they have recently used will not work, and that they must figure out how to solve this similar puzzle in a new way. To solve it, the player must employ a momentum saving strategy by attaching the Puzzle Ball to the anchor at the middle of the cylindrical cut-out and breaking the Tether right as the Puzzle Ball travels through that anchor. At that point, the Puzzle Ball would have the highest upward momentum, and would shoot up through the cylindrical cut-out and into the air above the cliff. Once in the air, the player must then remove the Chronosphere to keep the Puzzle Ball in the air while they travel to the top of the cliff in order to attach the Puzzle Ball to the anchor above the resting slot. Finally the player must reattach the Chronosphere to the Puzzle Ball to have it be pulled by the Tether to the final resting place to unlock the door.



Figure 14: The second part of the final puzzle.

This final puzzle presents the player with two similar structures in which they must move an object up, while also teaching them about using their two powers together. It challenges the player's knowledge of the momentum saving aspect of objects frozen in time as well as attach and detach the Chronosphere and the Tether in the proper order. By creating an easier puzzle to initially teach the player how to use the Tether to move the

Puzzle Ball up the cliff and then slightly modifying the same puzzle for the second part, our team was able to create a puzzle that challenge the player to stop and rethink their strategy. This puzzle requires excellent timing and dexterity to solve via removing the Tether and Chronosphere at the right times. However, the player can also take it more slowly by attaching the Chronosphere to the walls and floor to force it to stop upon leaving the volume. One strategy our team came across was placing the Chronosphere at the bottom of the cylinder cut-out so that the Chronosphere would automatically stop right as it got to the anchor point. From there, the player could just break the Tether and attach the Chronosphere to the Puzzle Ball to have it fly up well over the cliff.

The challenge of *Kairos Odyssey* comes from the puzzles. Since there are no enemies the player can fight, or any other dangerous elements, we wanted to treat the puzzles as antagonists the player faces. We wanted to make the puzzles ramp up in difficulty as the player progressed and end with a challenging puzzle that would leave them wanting more.

2.3.3 - Controls

The controls are minimalistic, so that they are easy to explain in-game and non-intrusive. *Kairos Odyssey* uses standard first-person keyboard controls: the W, A, S, and D keys moves the player and Spacebar lets the player jump. Moving the mouse rotates the direction the player is looking at. The right click shoots and returns the Chronosphere. Clicking and dragging with left click starts the Tether connection, and releasing it connects the Tether to the items selected.

Some alternate controls were considered, but not implemented. Raising and lowering the hand could be assigned to a key, perhaps Shift, or moved to the mouse and push the interactions to be activated by the Q and E keys. The benefit of this would be to increase immersion, give more weight to the abilities when they are activated, and show more of the environment on the screen when the player is not actively solving a puzzle. A run button was also a possibility, allowing the player to control their speed through the environment, so they can press and hold a key to increase their speed.

These alternate controls were not implemented in favor of keeping the most basic, standard controls. The controls in the game are the standard first-person controls, so players who have played that kind of game before will find them familiar. Another benefit of this control scheme is that all movement is on the keyboard and all interaction is on the mouse. This separation of movement and interaction is more natural to use than using the keyboard to control using the Chronosphere or Tether as well. Since the player will already be using the mouse to aim where they want to use their abilities, it makes sense to continue using the mouse to actually activate said abilities. The main reason to keep the controls simple is that the game is a puzzle game in which the player will be challenged to solve difficult mental puzzles. Having complex controls would add difficulty to our game in an unnecessary way, and we would rather keep all the challenge of the game in the puzzles.

As an extra note, this control setup would be easy to transfer to a controller by using two control sticks as the movement and camera control, and two shoulder buttons to use the primary mechanics. We briefly tested using a controller and found it to be effective and the game as a whole is ready to be translated to controller-based gameplay if we want to try that in the future.

2.4 Music and Sound

2.4.1 - Music

Like its visual counterpart, the musical component of *Kairos Odyssey* was crucial to establishing the tone and atmosphere of the game. One of the primary design philosophies of *Kairos Odyssey* was to create a cohesive and immersive experience for the player. This meant that the soundtrack had to embody the same sense of desolation and dramatic weight as the narrative without breaking from the texture of the game. To accomplish this, we knew right away that a traditional orchestral sound wouldn't be the right fit. This type of sound wasn't restrained enough for the atmosphere we wanted to establish with *Kairos*. Instead, the music needed a more natural, ambient quality without sacrificing a strong melody; a facet of many excellent video game soundtracks.

We created a two step music composition process -- first, write a song with strong melodic content that captures the mysterious and slightly foreboding mood of *Kairos*. Next, we rearrange and reorchestrate that song into a more ambient track that keeps the central ideas of the melody still intact. The result is ambient music that has some motion and interest to it while still not distracting the player from the gameplay and puzzle-solving.

2.4.2 - Sound

The sound effects of *Kairos* were carefully crafted to increase the player's immersion into the game world and provide visceral feedback to the in-game abilities. Many of the sounds for these abilities were created entirely digitally, but carefully crafted to avoid sounding 'tinny' or electronic. We also took care to avoid creating sounds that were overtly "magical" or reminiscent of typical fantasy "spell casting" effects.

The sounds for the Chronosphere took a while to get right. The base of the sound is formed from several synth patches made in Ableton Live. These sounds were then heavily edited and mixed within Reaper using a mixture of reverb, audio stretching, audio ducking to achieve the effect of entering and exiting the sphere. The sounds of exiting and entering give strong auditory cues to the player that time is either freezing or being unfrozen. When outside

of the sphere, the soundscape is a mixture of low volume ambient music and stretched ambient noise. However, upon entering the sphere, the game levels ambient audio track begins to play. The unaffected, organic sounds of the ambient gives a stark contrast to the sounds heard outside the sphere and provide a clear audio clue to the player that time is once again moving forward. The sound for launching the Chronosphere out of the gauntlet has a long build-up with a satisfying “whoosh” that helps give the action weight and significance.

The electronic-sounding Tether noises were created by mixing together various buzzing and electricity sound effects. The Tether has a steady buzzing sound that indicates its powerful pull without sounding grating or harsh. The Tether also has a finger-snapping noise for when it is broken. We recorded a few people snapping their finger and went with the loudest, most satisfying finger snap to make it enjoyable to use.

2.5 - User Interface

The user interface is intentionally very minimal. The only constant aspects on the screen are the hand and gauntlet on the bottom right corner of the screen, and a small dot in the center of the screen. The hand reminds the player that they are playing a role in this game, and the gauntlet provides all the status information they need to know. The availability of the Chronosphere is displayed on the wrist with a small orange sphere. Likewise, the availability of the Tether is displayed on the forearm with a small blue line.

The dot in the center of the screen indicates where the player is aiming, and also helps reduce the possibility of motion sickness³. Having all the status information on a model of a hand instead of on a more standard heads-up display (HUD) makes the game more immersive. By reducing the amount of constantly-present information on the screen low, the game is more minimalistic in this regard and allows the player to focus more on the environment and gameplay.

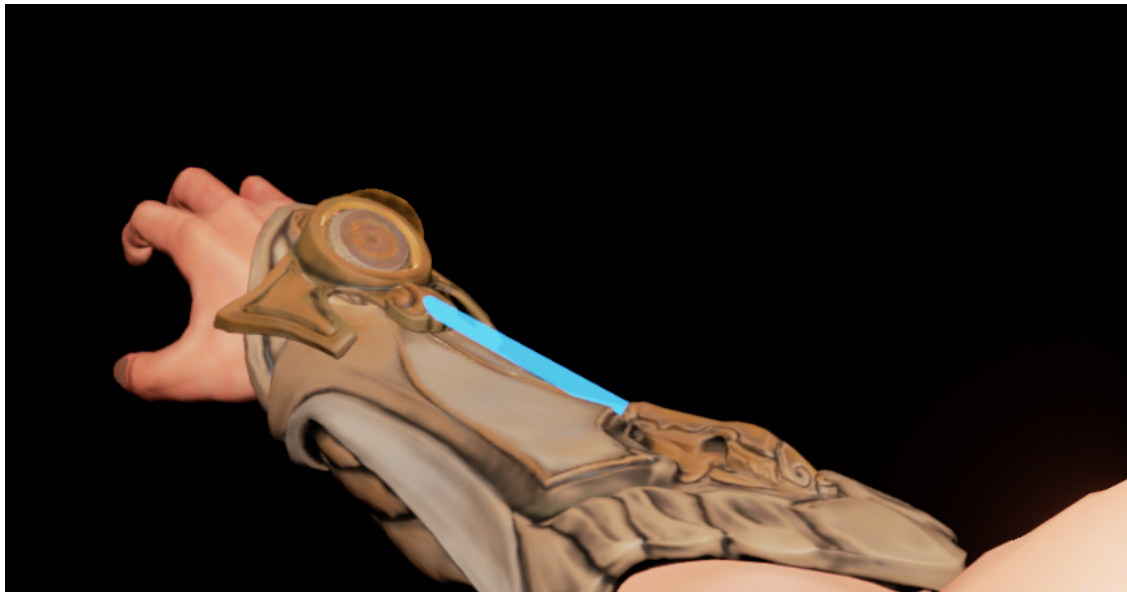


Figure 15: All elements of the UI displayed on the player's character.

³ "How developers are trying to solve motion sickness in video games".
<http://www.polygon.com/2013/10/26/4862474/video-games-and-motion-sickness-dying-light-techland-fps>

At one point we considered allowing the hand and gauntlet to leave the screen at times. When the player isn't solving a puzzle, they don't need to see the hand and gauntlet, allowing for even more immersion. This was decided against as it would require either setting certain areas as puzzle areas, or by adding a key to the controls for raising and lowering the hand. Also, the hand and gauntlet fit into the setting of the world and look appropriate enough to stay on the screen without breaking immersion.

3 - Implementation

3.1 - Scope

Managing to find the proper scope of our project was a very important task that our full team had to agree on before we began working on *Kairos Odyssey*. Our team had a lot of ambition, but knew that over scoping this project would result in a final product that was simply not done and not well polished. Because of that, we decided to aim for creating the perfect vertical slice of our imagined full game in order to create that polished game experience we wanted. We planned on having one level where we could display the main mechanics of the game and give a good idea of the direction that we wanted to take.

3.2 - Tools

3.2.1 - Engine

With a project of this scope, we knew that we needed to use a powerful 3D game engine to bring our ideas to life. We decided very early on in the project that we would be using Unreal Engine 4 to create this game. In our initial meetings, we debated using Unity, but eventually came to the conclusion that we could better achieve the visual quality we were aiming for by using Unreal. Unreal 4 is an extremely powerful and visually stunning engine, and it really helped us bring the world of Kairos to life. It's also quite new and very flexible, and we knew that understanding how to use it would help us greatly throughout our game development careers.

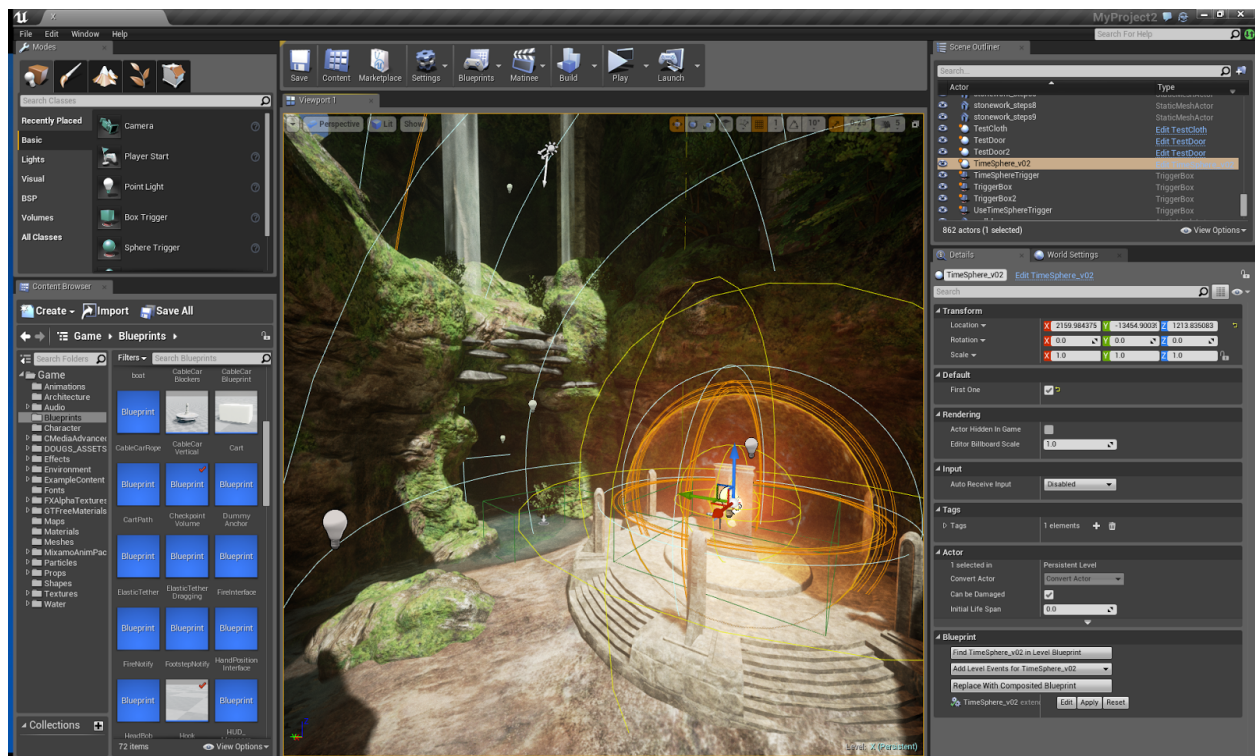


Figure 16: A screenshot of our finished demo in the Unreal Engine 4 Editor.

3.2.2 - Source Control

With a team of five all working on a project with a scope like this, there needed to be a way for all of us to be able to work on the project simultaneously, all while ensuring we were working on the latest iteration of the project. The best way to do this was by using source control, and the method of source control we chose was Subversion. The reason for this choice was twofold: Unreal Engine 4 has tools for Subversion already built into the engine, and WPI has a lot of server space to offer to Subversion hosted projects.

Unreal's method of handling source control was extremely helpful for us. Whenever an artist or programmer wanted to make changes to an asset, they would just have to check it out first -- which locks it from any editing from any other team member until you check it back in. The source control status of each asset is visible on the thumbnail in the Unreal Engine editor. Now the entire team could work on the game simultaneously, and our changes would never overlap or conflict since two people could never be editing the same asset at the same time. This also was a very nice way to know exactly what each member of the team was working on at any point in time.

3.3 - Workflow

3.3.1 - Asset Creation

The artistic asset creation involves a number of different components, which are interconnected in different ways. We categorized our assets into two groups: character model assets, and modular environment model assets. Even though most tasks can be performed with software like 3D Studio MAX, MAYA, Adobe Photoshop, and Zbrush, each group requires different workflow, so we assigned a group to each member of the art team based on their skill of using different software.

The first part of asset creation was creating concept art and thumbnails, which has the largest impact on the look and feel of the game. As we discussed in previous chapter, a large number of concept drawings were created using multiple references.

Beginning with creating a sketch and concept in Photoshop, we used the concept drawings as guides to create NURBS in Maya and then converted them to polygon meshes. Based on the game engine we used, we chose to convert into quads to match the NURBS model. The next step was box modeling and using combine and boolean operations to refine the base model. We managed to speed up the process overall by using existing models which we had created for other projects or came with the software package. For some modular pieces we used MAYA to create UV maps and bump maps, and then created diffuse and specular maps using Photoshop

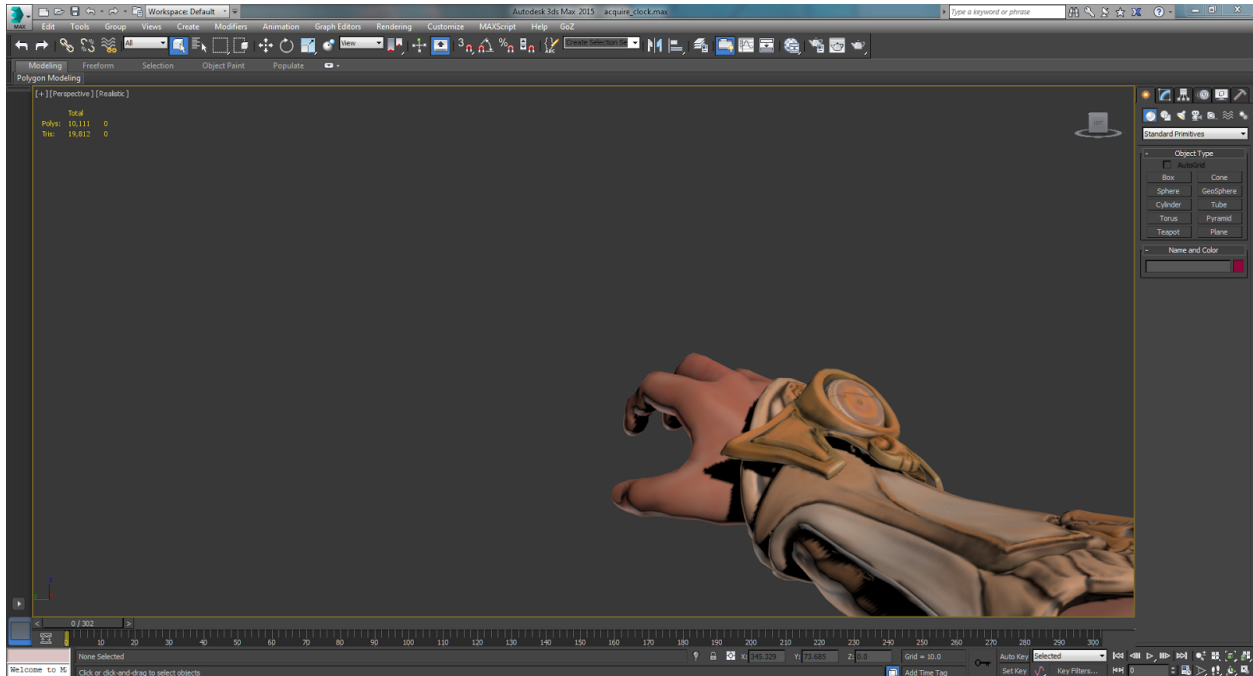


Figure 17: The completed character hand model in 3DS Max.

For the character model and hero object, we sent the base mesh to Zbrush using the GO-Z plugin. Working with different subtools and polygroups, we created a detailed high-resolution model. After that, we used polypaint and lightbox to apply a texture to the model. However, in order to make the model perform well in Unreal Engine, the polycount was limited to 10,000 for each object. Since some meshes had very complicated shapes, the combination of Zremeshing and project operation failed to provide a good topology. We ended up using Zspheres and re-topologizing the the model manually. Once the model was optimized, we unwrapped the model using the UV Master plugin and created a normal map and diffuse map in Zbrush.

Before we started rigging and animating the character, our advisor suggested us to test the model and texture in Unreal Engine. This step saved us a lot frustration later in the development process.

3.3.2 - Animation

To simulate the real life forearm rotation, we set up an automatic system that when rotating the wrist, the forearm will rotate a percentage of the rotation along with the wrist. Twist bone is applied to the forearm to achieve this goal, which creates bones to divide the forearm bone into two parts and uses wire parameter to wire the rotation of each bone to a percentage of the rotation given by the wrist bone. There are different approaches to get the twist bone in 3DS MAX, however we encountered a problem where the twist bone does not work with inverse kinematics. Due to this, we used a customized bone system instead of using biped or CAT.

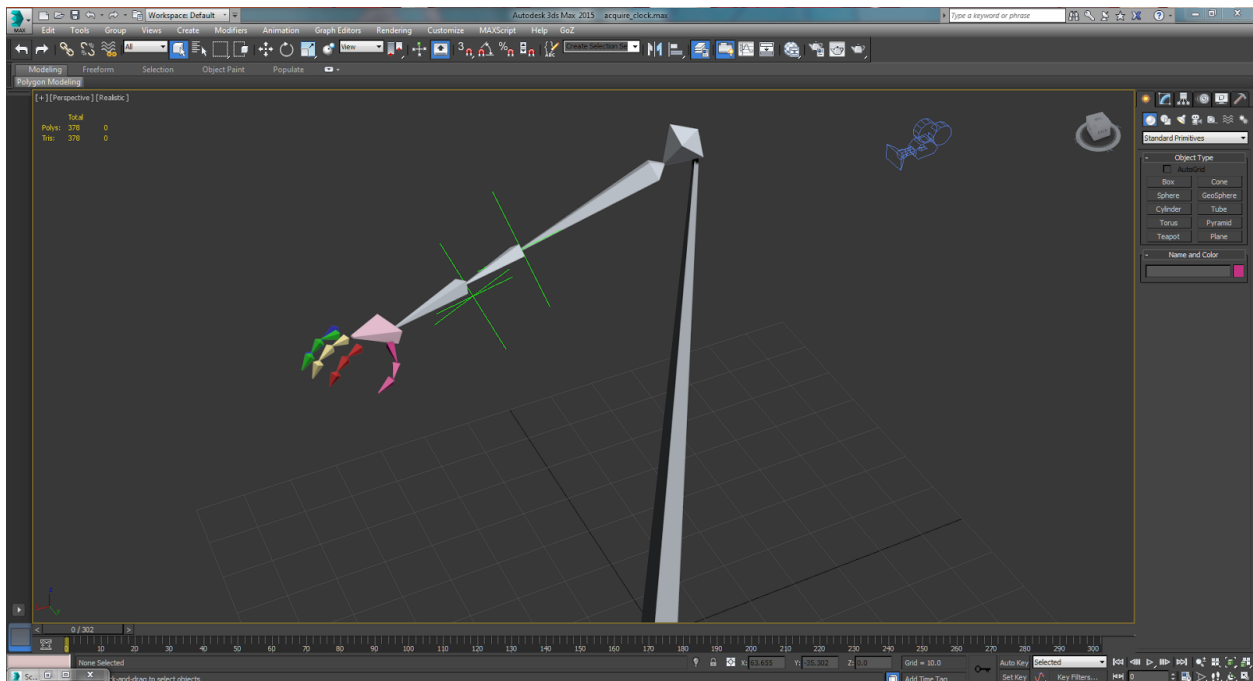


Figure 18: The bone structure of the player arm.

Considering that gauntlet is made of silver and gold, we linked the gauntlet to one of the twist bones so that there is no shape distortion of gauntlet when the wrist is rotated. Even though the arm model and gauntlet were skinned to the twist bones, we included the forearm bone in the skin modifier list anyway and the forearm bone has zero weight on any vertex.

When we animated the arm, we first set up a camera to simulate the first person camera in game. This way we were able to toggle the viewport between the camera view and perspective view to make sure the animation looks good in game. Many animations were created, including: idle, walk, jump_start, jump_loop, jump_end, chrono_sphere_start,

chrono_sphere_loop, chrono_sphere_end, tether_start, tether_loop, tether_end. For actions like jumping, shooting Chronosphere and shooting the tether, we created three states of each animation to achieve a smooth transition in game.

3.3.3 - Gameplay Development

Very shortly after coming up with our game concept and making the decision to use Unreal Engine 4, our team quickly went to work on a prototype. We had many goals for this prototype -- the first of which was to make sure our mechanics were solidly designed and, more importantly, fun to use. We also wanted to dive right into figuring out how to develop in Unreal Engine 4, as well as make sure our ideas were feasible to implement in the engine before going too much further into development.

After opening up Unreal's 3rd Person Shooter example and experimenting a bit, we determined the best way to make a rapid prototype in this engine was to use Unreal's blueprint visual scripting system: a complete gameplay scripting system based on the concept of visualizing program code as a series of nodes and connections. This graph of lines and boxes describes the flow of an object's (or a level's) scripting. Each node is basically representative of a line of code, and variables and arguments can be filled in by connections from other nodes. We liked this system as it allowed us to jump straight into prototype development, and it showed us exactly what Unreal was capable of doing. Just by looking at the list of possible nodes, we could see all of the functions the engine was able to perform.

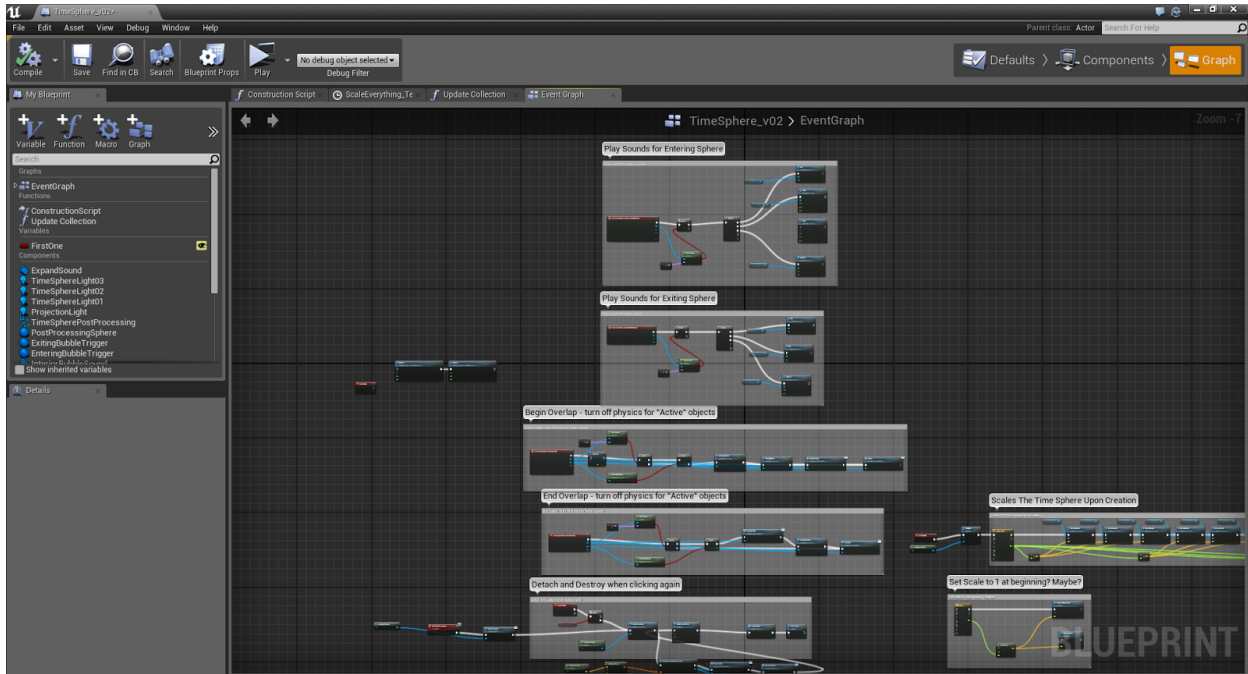


Figure 19: The Blueprint scripting for the Chronosphere.

The first mechanic we wanted to implement was the Chronosphere. This object was the key to our entire design, and we certainly wanted to make sure all of its functionality could be expressed in this engine. One of the first obstacles we faced was figuring out how exactly to freeze objects in time. After some brief experimentation with changing the time scale of various objects, we came to the conclusion that the idea of something being “frozen in time” can really be expressed in the engine by having physics not act on that object. Then, when that object is “unfrozen”, physics can once again act on it. So in the context of our design, we describe the Chronosphere as an object that unfreezes objects in time, but what it’s really doing in the context of the game engine is allowing physics to once again act on objects within its volume.

With this conclusion in mind, we went to work on implementing our primary mechanic. The initial implementation was actually quite simple -- every object in the world would begin without physics applied. Upon clicking the fire button, a projectile would travel in a straight line from the player, and once that projectile collides with anything, a Chronosphere would form at the point of collision. Physics would be activated on all objects within the bounds of the sphere, and turned off as soon as the object left the sphere’s influence.

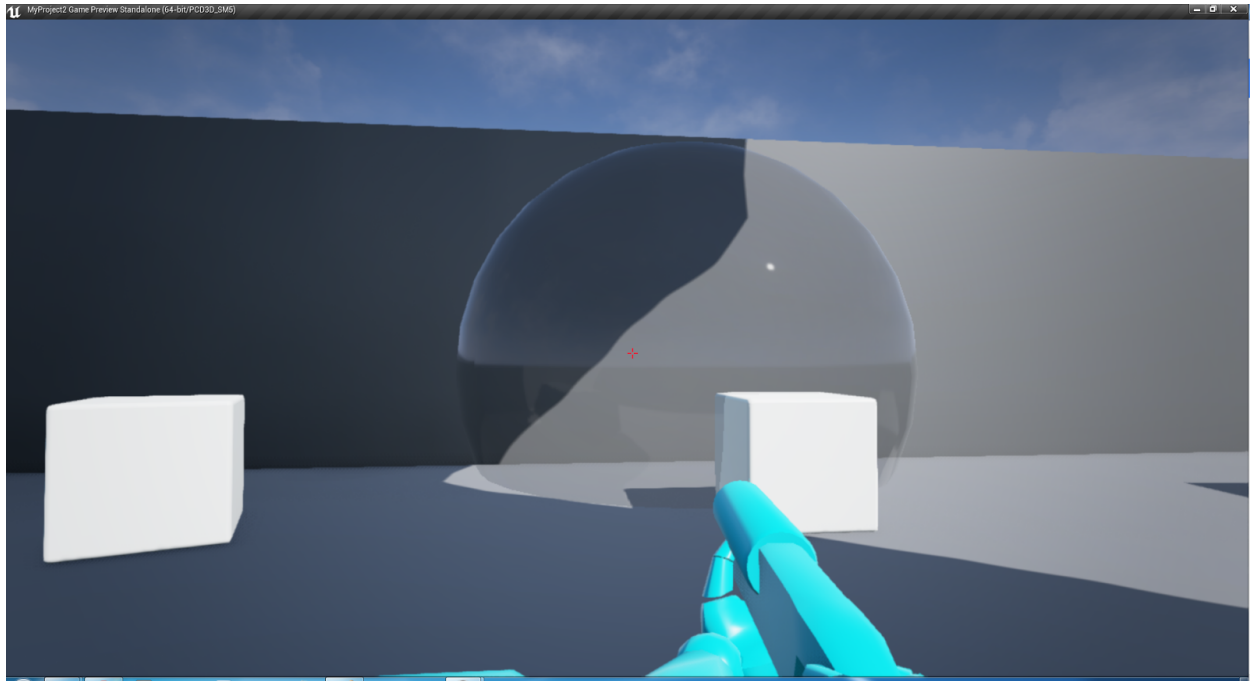


Figure 20: The Chronosphere prototype.

While this prototype worked almost immediately and was very fun to experiment in, we realized that we were missing a very important aspect of the Chronosphere. In this implementation, objects did not retain their momentum while being frozen and unfrozen in time. Instead, when time was reactivated on an object, it would behave as if it started from rest. To fix this, we made any object that would be affected by the Chronosphere have two local variables -- one to store its velocity, and one to store its angular velocity. These values were set to the object's current velocities just before it freezes in time. Once time resumes for an object, it takes these stored values and applies them to the object. This worked perfectly to make objects resume their past motion when being unfrozen in time. The variables and functions for doing this were stored in an interface, which is Unreal's method of communicating between various blueprints. If we wanted an object to be affected by the Chronosphere, then it had to implement this particular velocity-saving interface. This one change made the Chronosphere feel much more like it was manipulating objects in time rather than just turning physics on and off. Seeing a fast-moving object appear to pause on its path of travel and then resume motion exactly as it had before really solidified the concept of time manipulation.

Once we were happy with the implementation of the Chronosphere, we moved onto our other primary mechanic -- the Tether. We knew how we wanted the mechanic to operate: the player clicks on one object, then drags the mouse to another object while holding the mouse down, and releases the mouse button on the second object. Raycasting was used to find the points in the world where the player pressed and released the mouse, and it would add two invisible Tether reference objects at those positions and attach them to whatever objects existed at those positions. Then, force would be applied to each reference object to pull it toward the other object. The magnitude and direction of the force applied each frame to the objects is determined using the difference vector between the objects' two positions, resulting in each object being pulled toward the other.

The Tether underwent many modifications as we debated exactly how it should behave. Initially, the Tether could be attached to any surface, including floors, walls, and ceilings. As we changed our idea of how the Tether worked, so too did we change its implementation in the engine in order to test new possibilities. While the Tether's code changed dramatically over the course of the project, by the end we had returned to something very close to our original implementation. We added some editable constants to allow us to fine tune the Tether's behavior and power, and after much experimentation we decided upon a specific set of values.

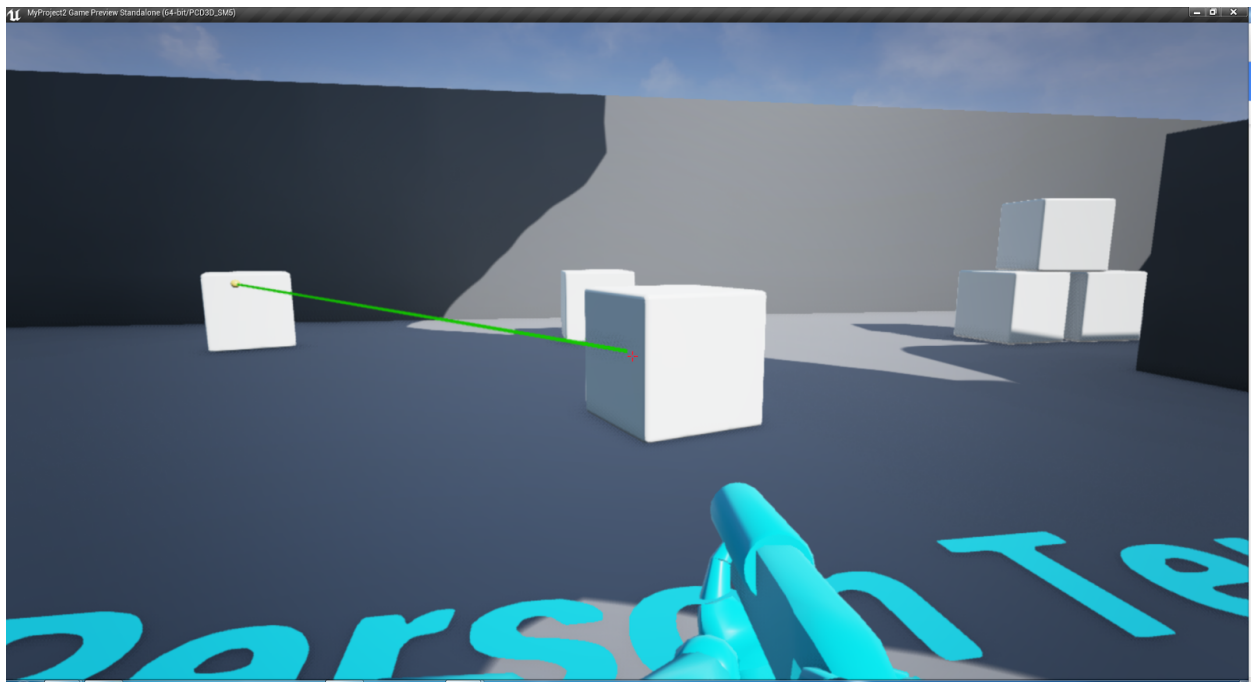


Figure 21: The Tether prototype.

One of the biggest technical challenges our team faced was in regards to Unreal's particle system. Since the world is frozen in time, we wanted particles to follow the same rules as objects -- stuck in place when outside the sphere, moving normally while inside. However, the in-editor tools for particle systems did not give us this capability. We could freeze all particles of an entire system, but there was no control over an individual particle's velocity. Instead of scrapping the idea, we decided to look to other methods to find a solution.

One of the nicer things about Unreal 4 is the ability to view and modify the engine's source code. We decided to create an additional particle module in the engine's source code that allowed particles to react to the Chronosphere, and apply it to any particle system that we wanted to obey the time rules of the game. Unreal's particle modules describe the behavior of particles on which the module is applied. Each module has two primary functions: an initialization function which allows changes to be made to the particle upon its creation, and an update function that is run on each particle in the system every tick. This latter function, the update function, was our key to solving this particle freezing issue.

We knew that in our module we couldn't make this function too complicated. Particle systems could have up to thousands of particles, and running a function on each one of those particles at 60 frames per second would create an immense load on the system. So instead of trying to do some complex code to see if the particle was inside the collision volume of the Chronosphere, we used a trick to speed things up drastically. Since the Chronosphere is a sphere, all we had to do for each particle was perform a distance check from the particle's location to the sphere's center point and see if it's larger than the radius of the sphere. The sphere's location and radius value could be passed to the system, and the update function could use those values for its calculations. If the distance check returns a value greater than the sphere's radius, then we set the particle's position to its last known position, resulting in it freezing in place. We also add that frame's delta time back to the particle's lifespan to make sure that any effects that take place over the life of a particle also pause and resume with its motion.

Before solving this issue, we were unable to use particle systems in our level designs because they would break the illusion of the world being frozen. But now that each individual particle also obeys the rules of the Chronosphere, we got the ability to really showcase how the the mechanic worked and add even more visual interest to our world. Unfortunately, this system was completed a bit too close to PAX for us to include it in our demo. However, we used a similar distance-checking trick to make the Chronosphere affect moving textures (such as water), a feature that we did manage to include in our demo.

Many of our early level designs included carts that the player could push along paths. For instance, before the player got the Chronosphere, there could be a cart with a Chronosphere on it that the player could push along a specified path to solve a puzzle. We wanted to create a system where a level designer could define a path and designate an object that could only move back and forth on that path. We found that the best way to define a path in Unreal 4 was to use a spline. Once placed in the level, control points could be freely added and edited. However, locking an object to this path was more challenging. The solution we came up with was moving the object's position further along the spline based on the angle of the push compared to the current angle of motion along the spline. If the push angle was closer to the spline direction, the object would move further along the spline. If the the pushing angle was offset from the spline direction, the resulting push would be weaker and the cart would move a smaller distance along the spline.

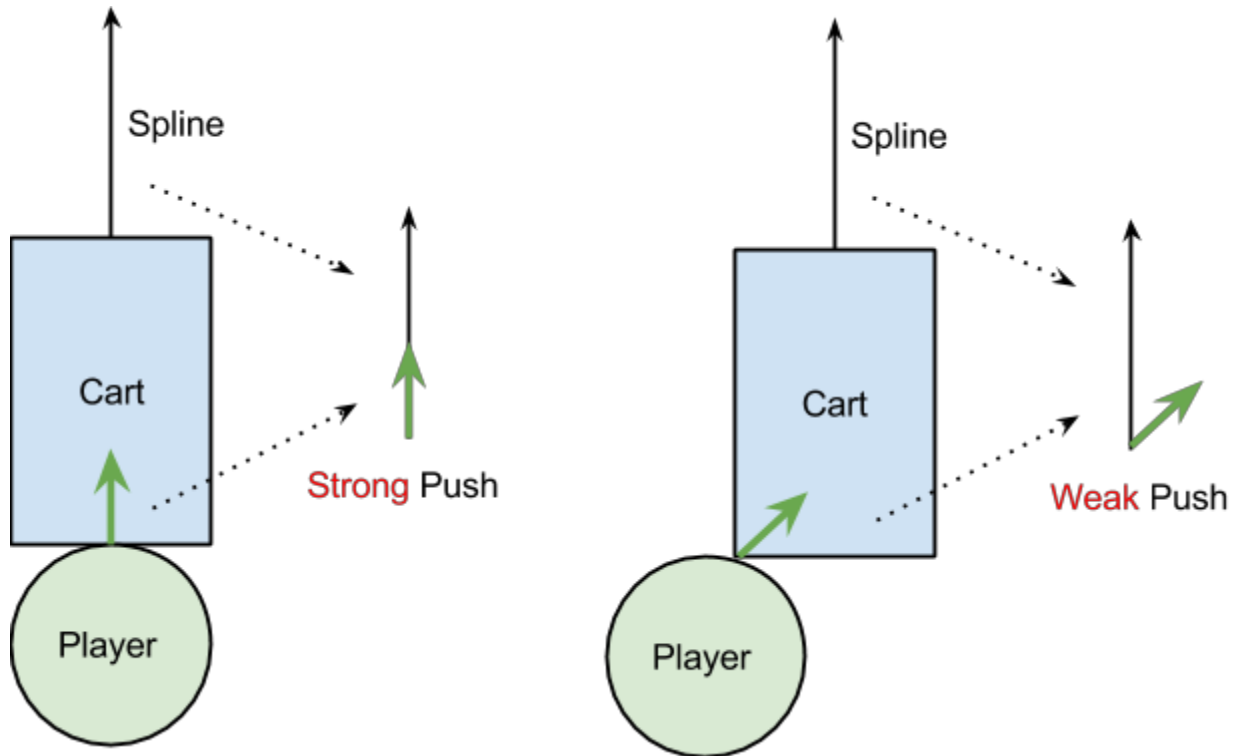


Figure 22: A demonstration of the cart-pushing mechanics.

This system ended up not making it into the playable version of our game, but the concepts here were used in the movable shrine that is featured in the introductory Tether puzzle.

While developing possible mechanics for the player, we came up with the idea for a sister power to the Tether. This power was the Rope. The Rope would be utilized like the Tether, where it would connect objects and anchor points together. However, the Rope would not apply any force to the objects it was connecting. Instead, the Rope would restrict movement outside of its length. With this, we planned to have situations where you would use the Rope to swing objects large distances, “monkey bar” objects across chasms, and swing moving objects around corners. Our most finished version of the Rope acted more like an iron bar than a real Rope, as objects were not able to move closer together and were constrained at the distance of the Rope at its time of creation. Despite the Rope not working as fully intended, it did manage to function correctly in the uses we initially planned for it. As the project continued, we scoped our level for PAX East and put Rope and any puzzles related to it at a lower priority. Sadly, we did not have enough time to add the necessary puzzles to our

level to teach the Rope and tie it in with the other two mechanics. Because of this we had to cut working on the Rope any further and focus on our higher priorities.

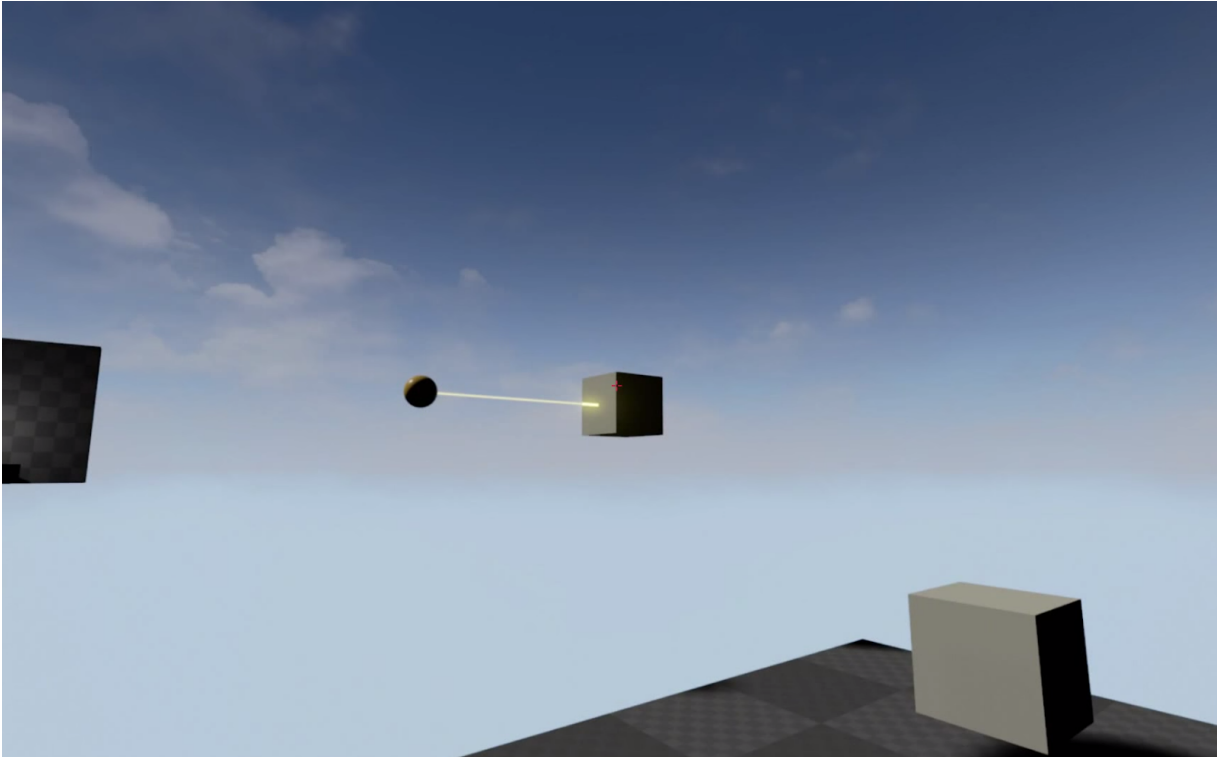


Figure 23: Our rope prototype, as seen in our PAX application video.

3.4 - Integration

Integrating the work of our Artists and Programmers into our actual game was a very important process that required proper communication to be effective. For our integration, we had our programmers primarily in charge of integrating the art assets into the proper game objects. These art assets were added by our artists into the Unreal project where our programmers could take them and apply them to the game objects and systems that they had created.

3.4.1 - Animation

When integrating our character's animations into our game, we had our artist in charge of the character and animations create multiple animations and then import them into the Unreal project. Once there, our programmers took those animations and added them into the

character's mesh. From there, our programmers could take those animations and add them to a state machine which would play those animations to match the corresponding state the player was in. This process was a very dynamic one, where our programmers would give feedback on how the timings and the feel of each animation worked out in game and our artist would respond with updated animations to match that feedback.

Outside of the animation state machine existed the montage system which could be called from an object to play a single non-looping animation. These montages were used for firing the Chronosphere and the Tether, as well playing the break the Tether and power acquired animations. In these non-looping animations, our programmers created notifies at specific times in the animations to better time events like the firing of the Chronosphere to their animations. This process of integrating our character animations was very efficient and saw almost no issues.

3.4.2 - Level

The integration of the level itself was handled differently than the integration of our other assets. One of our artists was directly responsible with creating the level with their own environmental assets. After they built the level, the programmers went in and began adding in the proper game objects in the right locations to the level.

Our team ran into two problems during the big integration of the level. These problems were lack of a set level layout and integration bottlenecks. Our team never created a set layout for the level and instead decided to give our environmental artists free control over the layout of our level after planning out the puzzles that would be in our level. This caused issues where our artist and programmers were not on the same page for where certain puzzles should have been and how the level was laid out. This could only really have been solved by the artists and the programmers communicating on their idea of the level's layout, but this process led to our second problem.

As time went on, we came across the issue of our progress being bottlenecked to the completion of the level. The level was being designed by our artist in a sequential flow, where

our artist would create the areas in order of when the player would reach them. Because of this, puzzles, assets, and systems had to wait for their specific area to be completed before they could be placed in the level. Our only solution to this problem once we were facing it was to add, test, and fine tune all the systems and puzzles that we would be adding to our finished level in a separate empty level.

Our team was able to work through those two problems and successfully integrate all our assets, objects, and systems into our level in time for our deadlines.

4 - Playtesting

4.1 - PAX

Kairos Odyssey was presented at WPI's booth at PAX East, from March 6 to March 8. The team members helped man the booth at different times throughout the convention, and close to a hundred attendees tried the game. It was incredibly helpful to watch people play our game demo and observe what they looked at in the scene and how they reacted to the situations they were presented with.

In between each day of PAX we updated our project to fix obvious problems and improve the experience as much as we could. There were countless preconceptions we had about our game that ended up being obstacles to players who started with no knowledge about the game. Each puzzle in the game was a significant obstacle to many players, which surprised us, as we expected nearly all of the puzzles to be simple to understand and easy to complete.

For the ball-rolling section, many players managed to knock the ball off the slope, making the puzzle extremely difficult to complete. Minute adjustments to the positions of the objects and some invisible walls helped prevent the ball from falling off so easily. In addition, a small force is arbitrarily applied near the lowest point of the slope to ensure that the ball makes the full distance into its destination.

In the final puzzle area, several Tether hooks were moved around to make the task easier and more straightforward to complete. The original hook placements allowed the puzzle to be solvable by several players but prevented many players who had similar but slightly different solutions in mind to be able to solve it. The new hook placements were more lenient while still requiring the same basic idea to be used.

Many players did not fully understand the basic mechanics, which is understandable given that they are only explained through clues in the environment. Before they need to use

the Chronosphere to proceed, the only direct hint to its ability is the ribbon flowing in the wind behind it. More hints of that kind would be very helpful.

The Tether tended to be better understood by many players, but some still had trouble understanding how to use it. The introduction to the Tether seems objectively much better than the introduction to the Chronosphere, but perhaps the Tether is slightly harder to use. Instead of just a single click like the Chronosphere, the Tether requires an accurate click and drag between two objects. Letting go of the mouse button before finishing the connection ends the interaction without connecting the objects, which seemed to confuse some players. To partially address this issue, the size of the hooks was increased to make it easier for players to connect a Tether to it.

Despite having a very small amount of text appear in the game, some visitors did not read the messages. As the text appeared and the gameplay stopped, they would use that as a break to look away from the game. This was possibly due to the exciting nature of the convention, but also a very observable demonstration that many game players avoid reading any text in games. The text in our game is possibly not needed, other than explaining controls.

5 - Post-Mortem

The following sections are individual accounts written by each team member. Instead of giving one global opinion, we gave each team member a chance to record their thoughts.

5.1 - Doug Davis

My primary roles in the development of *Kairos Odyssey* were as lead artist and 3d modeler. Much of my work involved modeling various 3D environment assets within Maya and integrating/assembling them into our levels in the Unreal Engine. Throughout the project I became very familiar with box modeling and poly modeling techniques in Maya. I had to develop a large array of modular, organic and architectural assets, as well as some large set piece objects for use in the game. In addition to modeling, I also had to create uv, texture, normal, and specular maps for each object. These were created through several means including hand painting them in photoshop, personal photography, or altering images found online that I couldn't acquire myself. Normal and specular map generation was done primarily using the software Crazybump.

In the Unreal Engine I spent a lot of time working with the material editor to fine tune the textures for each object. I also created materials and effects for certain FX elements of the game such as the undulating surface of the Chronosphere and the water/waterfall effects. Finally, I spent the last couple terms of the project assembling the levels for our game. This involved working with a variety of modular assets and creating the caves, valleys, and vistas that the player encounters throughout the game. I worked carefully with lighting and post-processing effects for each area to each achieve the right amount of visual drama.

In addition to these roles, I also worked closely with Andrew on the music for the game. From the beginning, we both had a very clear idea of what we wanted to achieve with the music; a hybrid of muted piano instrumentation weaved into somber ambient electronic tracks. The music of *Kairos Odyssey* was a great example of being to use our distinct styles and skillsets into a cohesive whole.

Working on *Kairos Odyssey* was a creative joy for me. I found the project to be artistically inspiring, and it allowed me to test my abilities in a variety of ways I never had before. I can't understate how important the Unreal Engine was in this aspect. As someone

who is not familiar with programming, I found Unreal's visual scripting to be incredibly powerful and easy to use. It allowed me to very quickly conceive and implement a host of technical art features that would have taken a complex understanding of coding to create in another engine. My favorite example of this is the Chronosphere; I was able to precisely recreate my original concept within the engine, complete with the undulating, smokey surface and projection light that casted rippling caustics onto the surrounding walls. Throughout the development of *Kairos Odyssey* I gained a huge amount experience with the development pipeline of creating an actual game. Though there were definitely stumbles along the way. Looking back on the project, our biggest pitfall was the difficulties we faced combining our level and art assets with the actual puzzle elements of the game. This was a problem mostly of the amount of time it took to create each of the in game assets. I think that in future, this problem could be remedied by creating even more fleshed designs. A dedicated concept artist for the whole project would helped create more structured asset and puzzle creation.

5.2 - Mark Foster

In this project I had a role in the tech team, which involved implementing the gameplay and solving various issues. Much of the work was done alongside the other members of the tech team, Andrew and Chris. I worked with them to implement the game's Chronosphere and Tether mechanics, to design levels and puzzles to make use of those mechanics, and to bring the demo to a playable state. I was involved in working in Unreal Engine's source code to create the custom particle module. To some degree, I opted for the tasks that seemed like no-one else wanted to do, so that the other members of the team could focus on making excellent content. I set up the source control we used, initially created the title screen, and made the lock-key-door system. I also assigned myself to fix a few difficult bugs that had been ignored for a while, such as a loud collision sound bug and an edge case where the Chronosphere wasn't being removed correctly. In our group discussions, I tried to remind the team that we needed to actually create a game from the ideas we were generating.

The Chronosphere is an amazing idea, but it was incredibly hard to precisely demonstrate its rules and functions. But only a few months ago, that's what we set out to do. My one main complaint about this project is that a cave setting turned out to be a poor choice

for trying to convey that time is frozen. For projects before this MQP, I had gotten used to being able to make design decisions solely based on the best choice for gameplay, but we had such a focus on aesthetics that this was no longer possible. It was difficult to make such concessions, but I have to admit that the end result turned out very well.

5.3 - Kedong Ma

As a junior student, my roles in the group were concept artist and 3D modeler. I was also in charge of rigging and animation. Working on Kairos Odyssey allowed me to apply everything I have learned from previous classes such as concept art, 3D modeling, animation, and rigging. I am glad that I went through the entire game development with my team. It helped me define my concentration in my future career as a technical artist, and really encouraged me to move on and plan more ambitious goals for my senior year.

I didn't successfully set up source control until D term. This meant that I couldn't directly engage myself with the entire development process or implementing my art assets in Unreal Engine. After I set up source control, I was able to manipulate the levels using my own computer and have a better understanding of our game mechanic. Although I finally created three new levels design based on our game mechanic with other team members, it was too late to update our game with those cool ideas. Even though we can still implement those ideas in the future, we would have a more polished game if I was able to set up the source control at beginning.

5.4 - Andrew Strout

My role in the group was lead programmer and composer. I took the lead on working with Unreal and implementing our mechanics and systems. I set up our initial implementation of the Chronosphere and the elastic Tether early on in A-term, and continued to tune and tweak them throughout the course of the project. I also created systems for carts that travel along paths, mushroom platforms, a custom particle module, and many other assorted small

systems and mechanics. I also created an original song for our game demo. Also, like all of the other members, I had a hand in the discussion and design of our game.

I think many things went right with this project. We brainstormed an excellent original game idea and were able to completely demonstrate it in three terms, which is no small feat. I'm very proud of how our mechanics feel and operate in the game; it was a very faithful representation of our original vision. Another aspect of the project that I think was very successful was our choice of engine. Unreal allowed us to create, test, and iterate very quickly. The initial prototype only took about three days to put together, and that was before we even really knew how to use the engine. I think the choice to use Unreal over Unity was an excellent one, at least from my personal standpoint. I found the engine to be more powerful, better organized, and overall less confusing than its counterpart.

But of course, Unreal wasn't perfect. Looking back, there are a few things I would have done to smooth our development process a bit more. At one point in development, all of our mechanics and systems broke due to an update to the Unreal Engine, and I was forced to reprogram the Chronosphere and Tether mechanics from scratch. Thankfully, this happened early on in the process, so there wasn't too much to correct. If that happened later on in the process, it could've eaten up a great deal of valuable development time. The lesson learned here is that, at least until Unreal is more stable, freeze which engine version you're using once development starts picking up speed.

The only other sizable problem we encountered was a rather stifling bottleneck in our level creation pipeline that manifested itself just before our PAX demo was due. We had plenty of mechanics and systems ready to go, but we needed our level graphics to be completed before we could implement them. Unfortunately, work just piled up on the artistic side of things. I think this is due to our very high visual standards for the game. Creating assets that were up to our standards unfortunately took a very long time. The only way to solve this problem without compromising quality would have been to have another member on the art team. It was just a bit too much work for two people.

5.5 - Chris Turner:

My main role in the development of Kairos Odyssey has been as a programmer with a side role of level designer. I worked with both Andrew and Mark in creating the various gameplay elements and systems we incorporated in Kairos Odyssey as well as working on creating designs for implementations of puzzles in our game's level. Early in our first term of the project, I worked on the Chronosphere and the Rope features with Andrew, then focused on creating different puzzles which we could implement using the mechanics we were creating. Towards the end of our project I worked on creating a custom particle system module with Andrew and Mark which would allow for particle systems that would properly flow inside the Chronosphere and freeze while outside. I then focused on creating a system for on screen message display that we could use throughout our level to instruct our players. Finally I took part in final level alteration before our game's showcase at PAX East.

I feel that our team has been very strong throughout the past year of working on Kairos. We got together right before the end of the 2014 school year to discuss and try to pin down the type of game we were making, and had very productive design talks. Going into the start of our MQP, we already knew what type of game we wanted to make and what we would be working on.

Overall I felt like our project went very well, but looking back I see two things that I wish I could have changed. The first of which was my work on programming mechanics and planning puzzles that ultimately became unused. All the work on floating platforms, designing puzzles and uses for the Rope, and work on the Mushrooms with interaction with the Tether felt wasted when we decided they were out of scope for our PAX East build. The second would be the issues that appeared before our PAX East showing where our pipeline became extremely backed up and caused a bottleneck of our progress of combining our programming changes without latest art assets. I personally feel like I should have taken a larger role in the level design to have alleviate this issue, but we were able to push through these problems and deliver a game to PAX East that I felt we felt proud to show off.

I am very proud of everything our team has been able to accomplish in this past year and I think we have really created a great vertical slice like we planned to do from the start. I am extremely happy that our team also plans to continue working on *Kairos* past the scope of this project and I can't wait to see how far we can take it.

6 - Conclusion

We consider the *Kairos Odyssey* MQP to be a success. We set ambitious but manageable goals for our project that we were able to meet. Our advice to future MQP teams is to understand the team's capabilities, scope your project appropriately, and select the right tools. Unreal Engine 4 was absolutely the proper choice for *Kairos Odyssey*.

Creating *Kairos Odyssey* was an incredible experience for our entire team. We are very proud of our finished playable demo, and it is very rewarding to see how far the project has come since we brainstormed the initial concept. As we mentioned, this is just a vertical slice, not a completed game. We would like to continue working on *Kairos Odyssey* and create the completed game, per our original scope. Our demo was very well received by playtesters, many of whom expressed interest in purchasing a full version. We had a very talented team who worked well together, and our team members are all inspired to continue with the project and bring *Kairos Odyssey* to its full potential.

6 - Appendices

6.1 - History of the World Clock

MQP Project Kairos: History and Backstory

Long ago, the Gods built the world and created beings to inhabit it. These mortal creations, however, were lost without the ability to perceive time and space. The Gods, who are immortal and infinite and thus have no need to perceive time and space, built the World Clock and placed it in the mortal world to give the beings a way measure their lives. They set the clock into motion, and from that moment on the beings could perceive their existence in the world, and time was created.

However, as centuries passed the beings began to question their existence, their power and their place in the world. They believed that if they could understand and control the World Clock they could become Gods themselves. Eventually they climbed the tower walls, hoping to dismantle the immense clock face built into the monumental structure, not realizing that in doing so they would destroy the Clock Tower, releasing its power in a catastrophic wave of energy and freezing time to a halt.

In punishment for their transgression, the Gods left the tower unrepaired and tasked the remaining beings with the responsibility of restoring power to the broken structure every hundred years. Each century a being is chosen for the task of delivering the source of power to the structure, during which he must prove his worth and devotion to the gods in a gauntlet of trials.

6.2 - Original Pitch Document

MQP Project Kairos (working title)

Concept Overview



Cave Level Concept

Project Group

Douglas Davis – 3D modeling, Animation

Mark Foster – Programming, Level Design

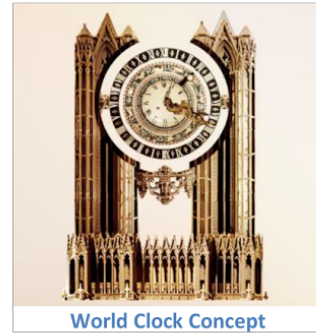
Kedong Ma – Concept Art, 3D Modeling

Chris Turner – Programming, Technical Artist

Andrew Strout – Sound Design, Composer

Game Summary

Project Kairos (temporary title) is a linear, first-person puzzle/platforming game set in a desolate landscape that has been frozen in time. In *Kairos* the player controls an unnamed protagonist who has been tasked with restoring power to the World Clock, a massive monument built by the Gods and placed in the world so that mortal men could perceive time. Every 100 years the World Clock shuts down, freezing time, and a chosen hero must deliver a new power source to the structure. (A document explaining the story of the World Clock has been included with this one)



World Clock Concept

Throughout the course of the game, the player will journey through a series of trials designed by the Gods to test the one chosen to repair the World Clock. This journey will take the player through a dark, underground cavern, up into the mountains and, finally, to the very face of the world clock.

Gameplay

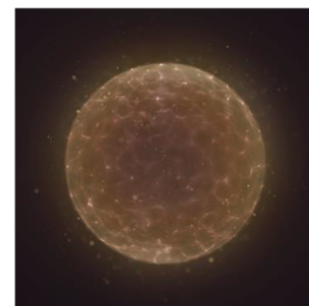
The gameplay of *Kairos* revolves around first-person puzzle/platforming challenges. The player can move, jump, and grab/hold objects. However, the world of *Kairos* is frozen in time. All tangible objects are immovable, frozen in place at the moment the clock shut down. This provides a unique and interesting dynamic to the challenges presented in *Kairos*. To solve these challenges, the chosen hero uses a special gauntlet with two unique abilities.



Gauntlet Concept

The Gauntlet's first ability is the **Gravity Chain** (name subject to change). The player uses the Gravity Chain by marking two objects in succession, A and B. Once the second object is selected, the two objects will be pulled together with by an invisible, elastic force. This ability can be used to lift heavy objects, move platforms, hit targets etc... However, objects cannot be moved with the Gravity Chain until they are unfrozen with the player's second ability.

The **Chronosphere** is the second ability given to the player. The Chronosphere is a projectile casted from the player's gauntlet that, upon landing on a surface, expands to create a large bubble (about 10 feet in diameter) in which time moves forward. This bubble will stick to the surface it lands on, allowing objects to be moved, destroyed, or manipulated in various ways. It is important to note that any forces that were acting upon the objects when frozen are retained and will continue to act when they are unfrozen (a thrown ball will continue its trajectory when unfrozen).



Concept for Chronosphere

These two abilities can be combined to create a variety of gameplay challenges, most which will be based around carefully setting up the gravity chain and then using the Chronosphere to execute the setup.

Setting

Project Kairos takes place in a harsh and desolate region of an alternate world. The World Clock, and the area surrounding it, is completely devoid of civilization, other than a handful of ancient ruins left behind



Location Concept

from before the World Clock was broken. The hero has had to travel far from home to reach the area. The landscape is harsh, and organic, ranging from barren tundra or desert, to frigid towering mountains. Throughout the game the player will venture through a series of underground caverns, rocky canyons, and, finally, the top of the World Clock structure.

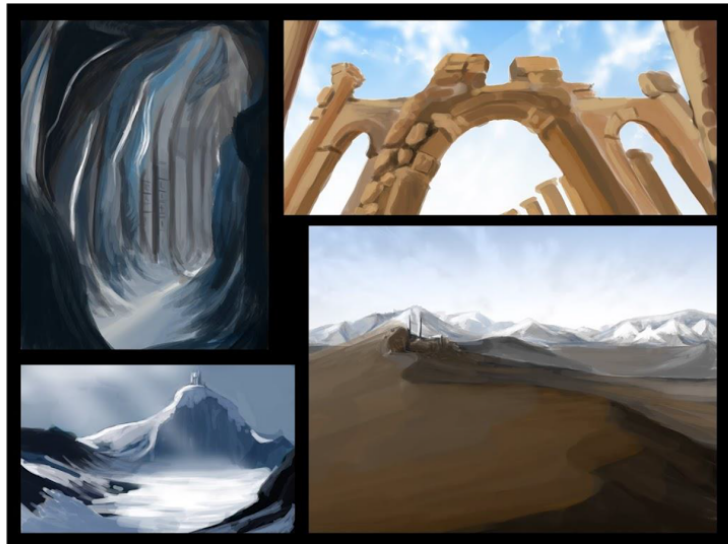
The visual aesthetic of *Project Kairos* is very important to us. We want the player to feel completely immersed in a believable and natural world. To achieve this, we are designing the puzzles and

gameplay challenges to feel like natural extensions of the environment. In one section, the player may have to use his abilities to maneuver a small boat through a dimly lit cavern, or to repair a broken bridge. We don't want anything the player encounters to feel too 'game-y' or contrived.

We also want the world to look beautiful, both graphically and artistically. Ideally, we want to use a combination of baked light mapping and real-time lights to achieve a high level of visual fidelity without sacrificing too much performance. Much of the challenge of creating the game will be from creating the many high-quality 3D environmental assets that are needed.



Cave Level Concept



Location Concept

Project Goals

With *Project Kairos*, we really want to emphasize quality over quantity. We hope to provide an experience that is beautifully presented, immersive, and engaging for the player. We have made deliberate efforts to keep all aspects of *Kairos* simple and elegant, so that everything that IS in the game is executed with a level of polish akin to an AAA game title.

The story is simple, yet compelling for the player. It also provides a game world that is isolated and uninhabited, meaning we won't have to spend time on complex character models and animations.

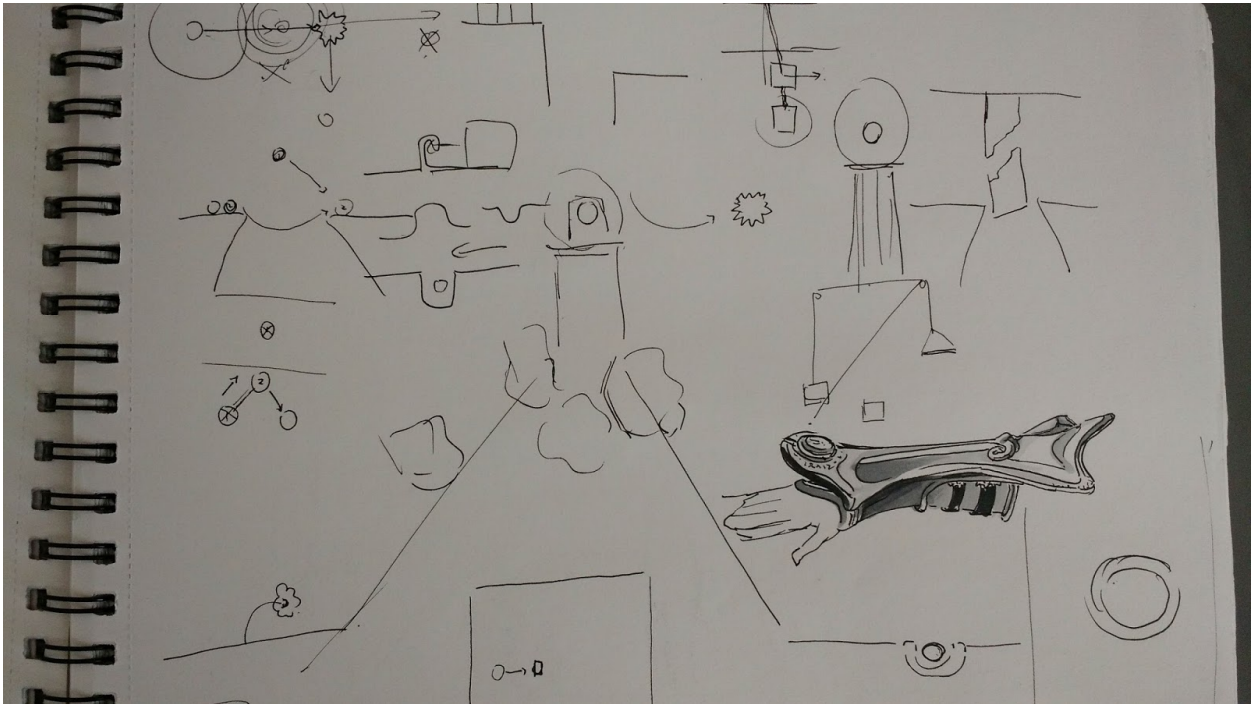
Level design will be kept linear and controlled, allowing us create puzzles that are interesting, complex, and fit with the aesthetic of the game.

We have deliberately chosen to only give the player two unique gameplay abilities, so that the execution of these powers (the way they 'feel', in a visual and audio sense) is exciting and rewarding for the player. We believe that sticking with two relatively simple mechanics will allow us to provide depth to the puzzles, rather than breadth.

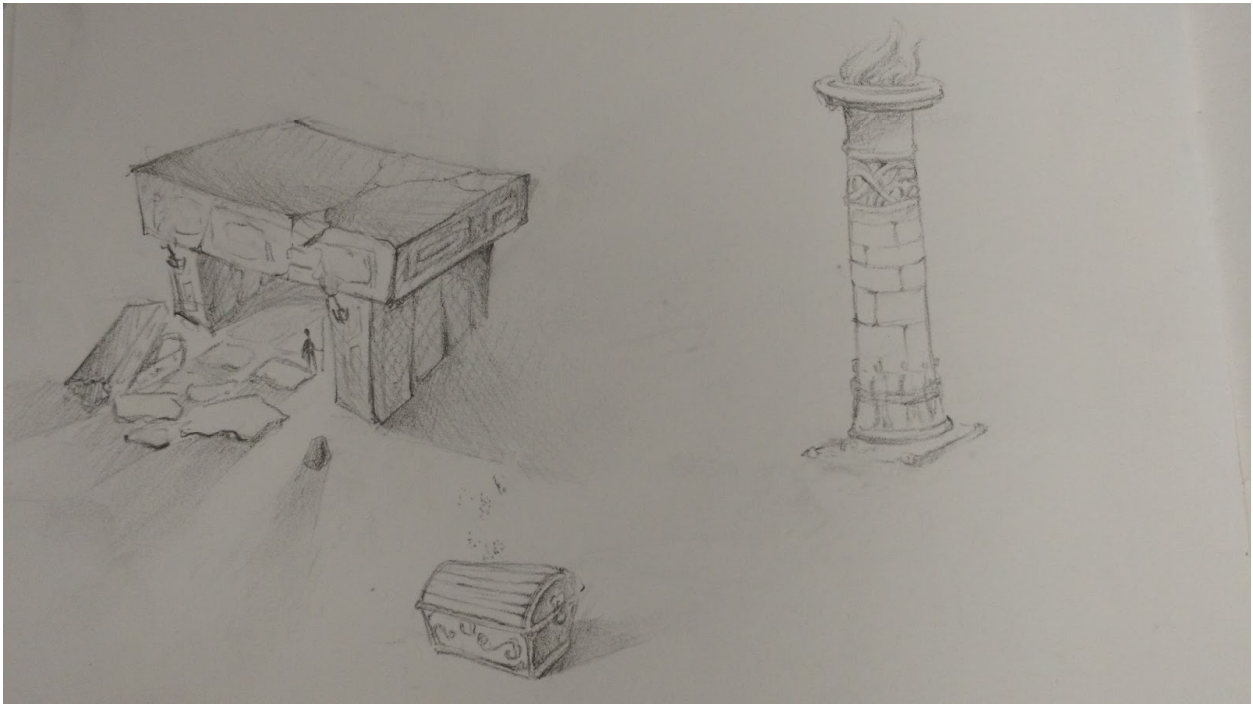


World Clock Concept

6.3 - Notebook Samples



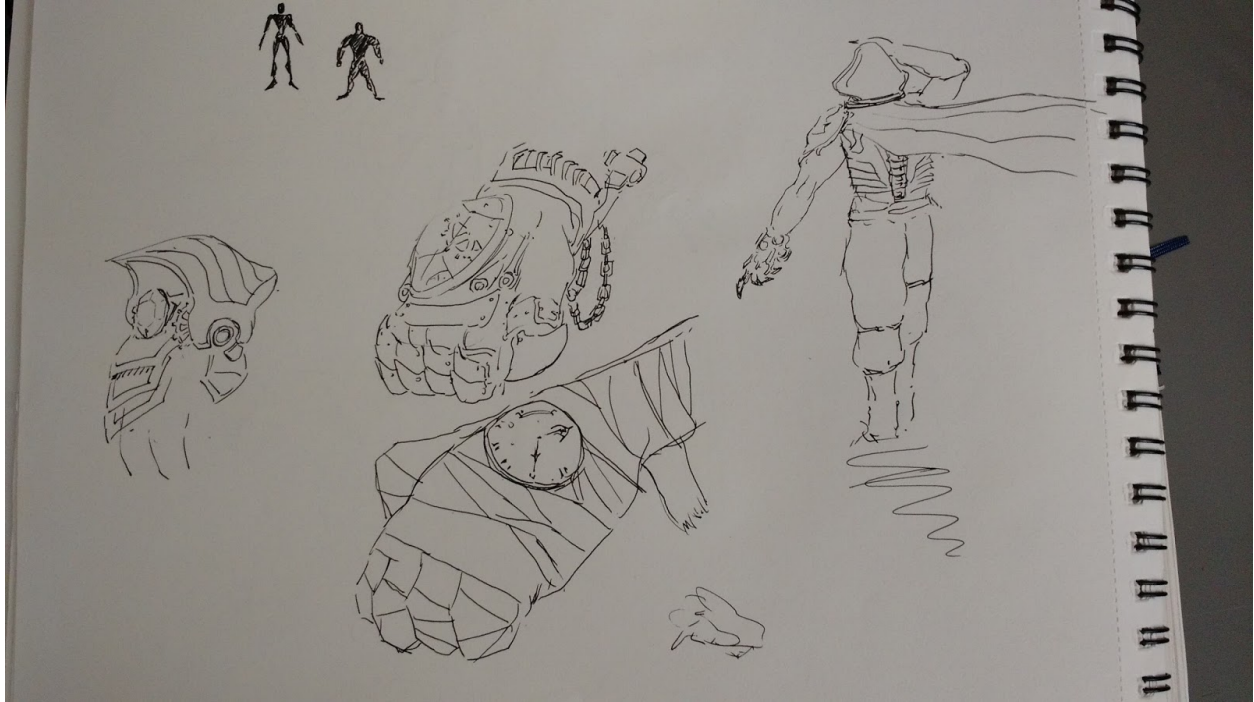
Some initial puzzle ideas and early gauntlet design.



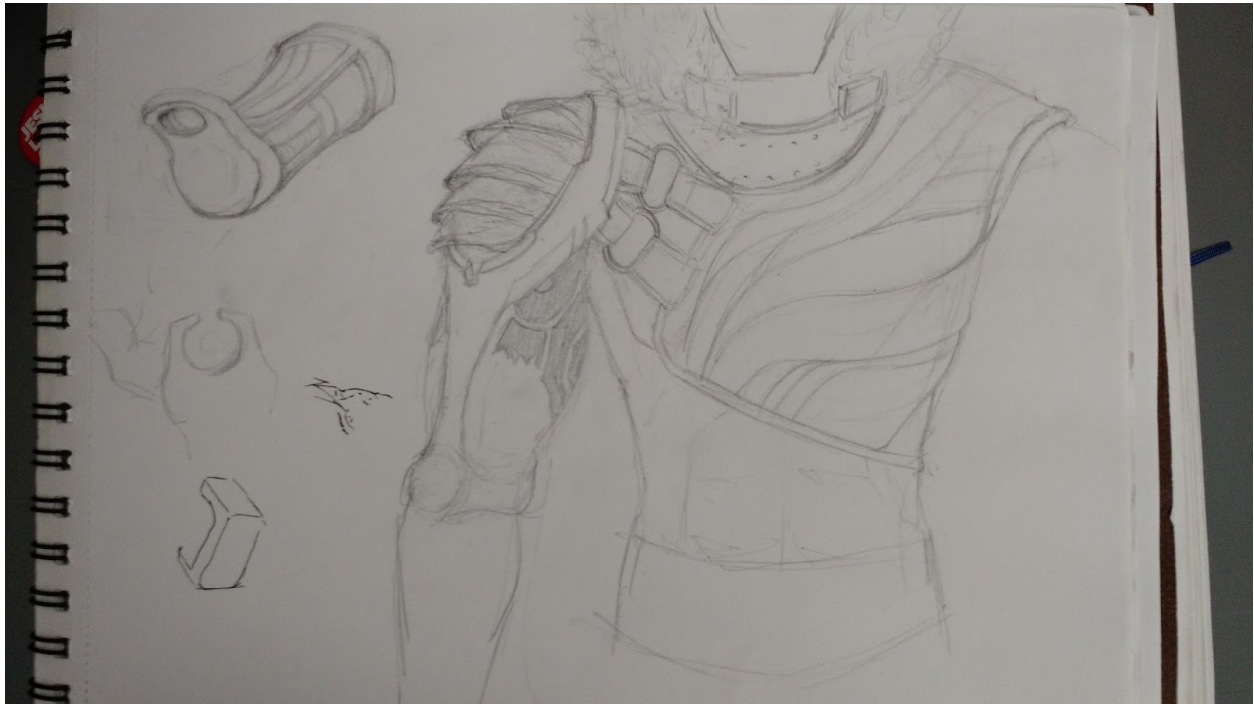
Environmental piece concepts.



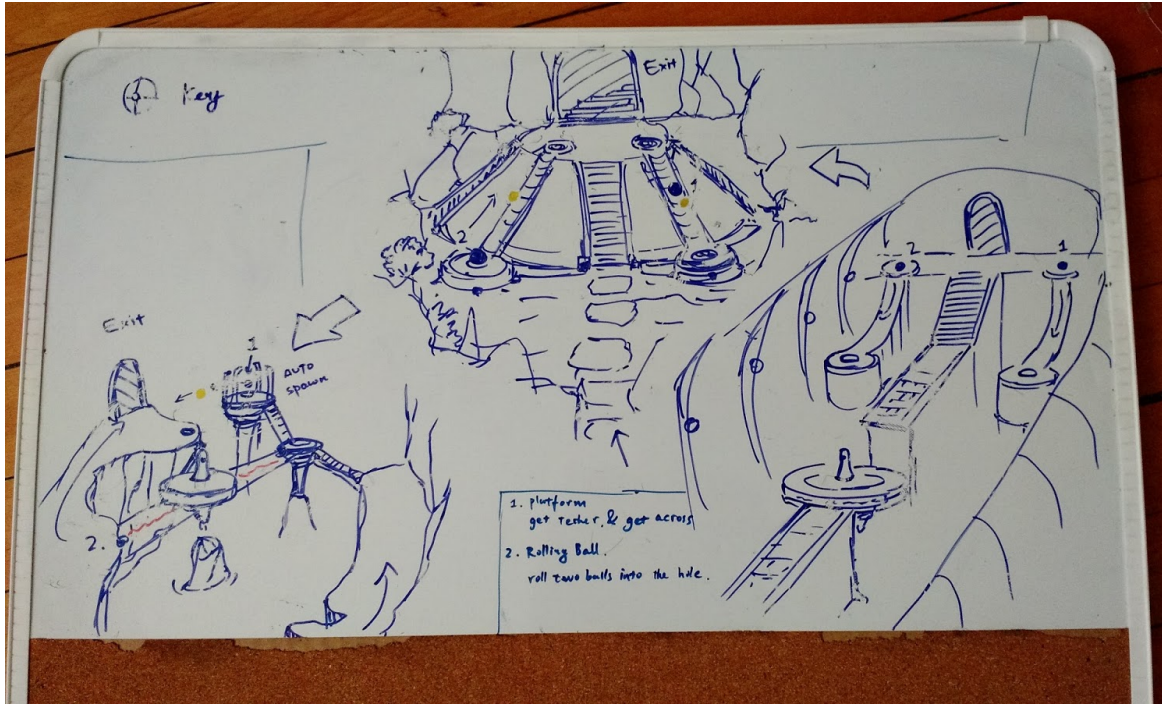
Early character concept.



More character and gauntlet design sketches.



Outfit and shoulder armor concept sketches.



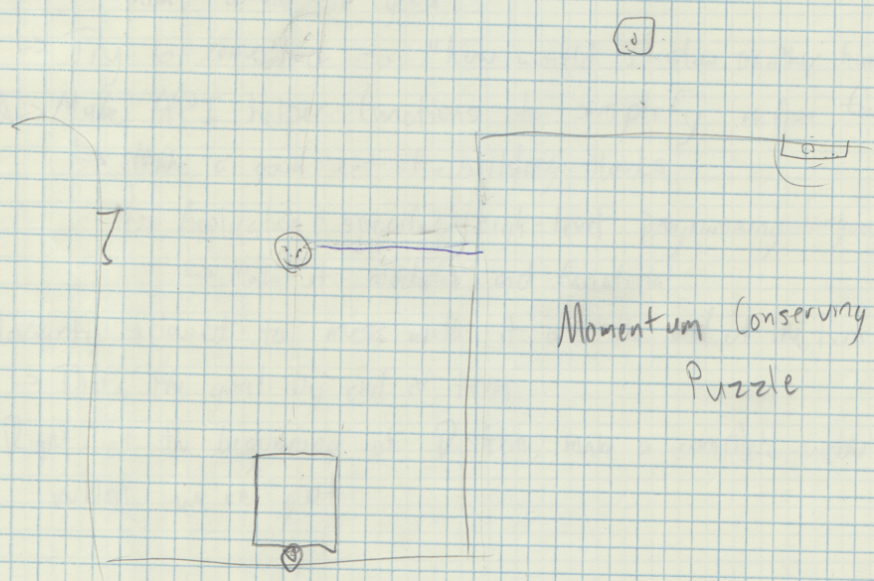
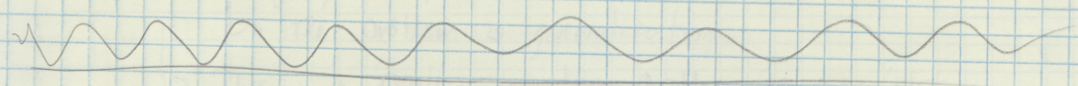
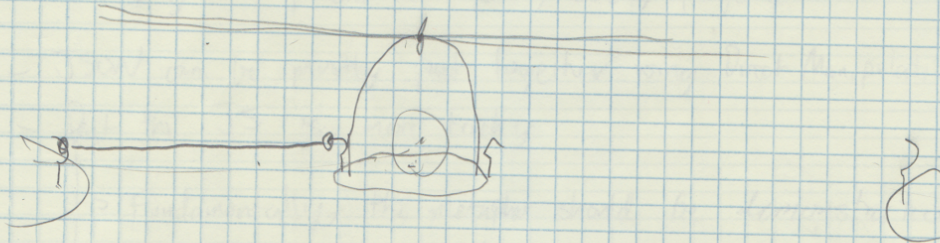
Level design ideas involving rolling ball and platform puzzles



Some puzzle ideas involving rocks, hooks, and mushroom platforms.

- Cable Car Puzzle

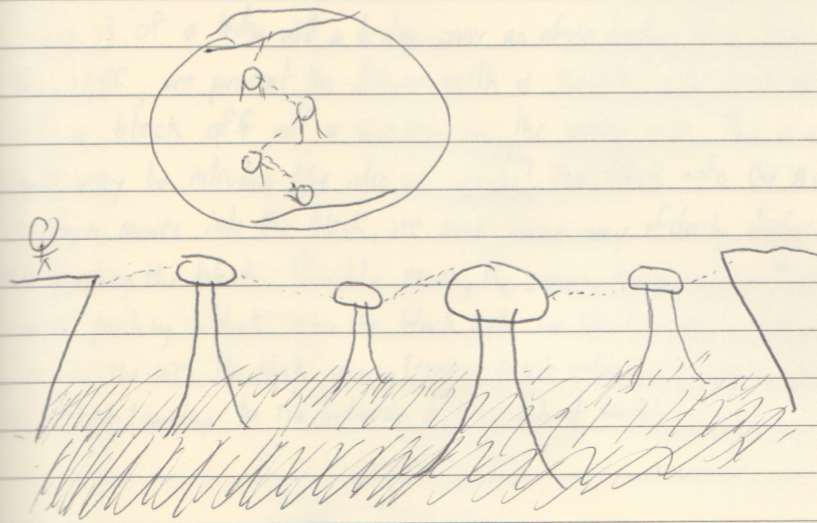
- ↳ Showcases tether mechanic
- ↳ Forces player to use it to get to other side



More puzzle ideas, including the Tether's introduction puzzle.

Mushrooms!

Mechanic: Grapple



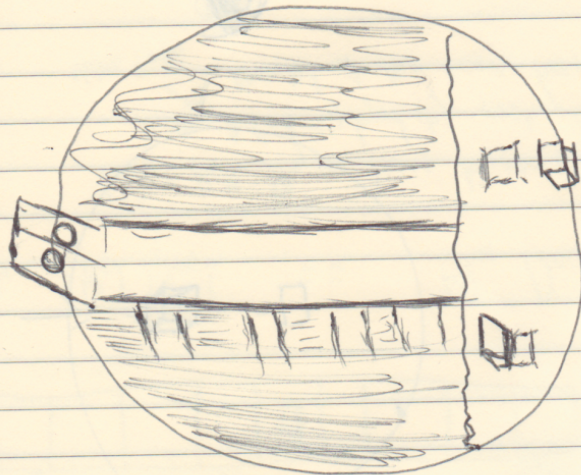
For the grapple, we want to show that you can move objects so that you can utilize them, and that gravity is not the main source for solving puzzles. We will have pillars across a gap that are constrained to the ground. These pillars will be able to be pulled closer together by the grapple to allow the player to traverse them.

More mushroom platform ideas.

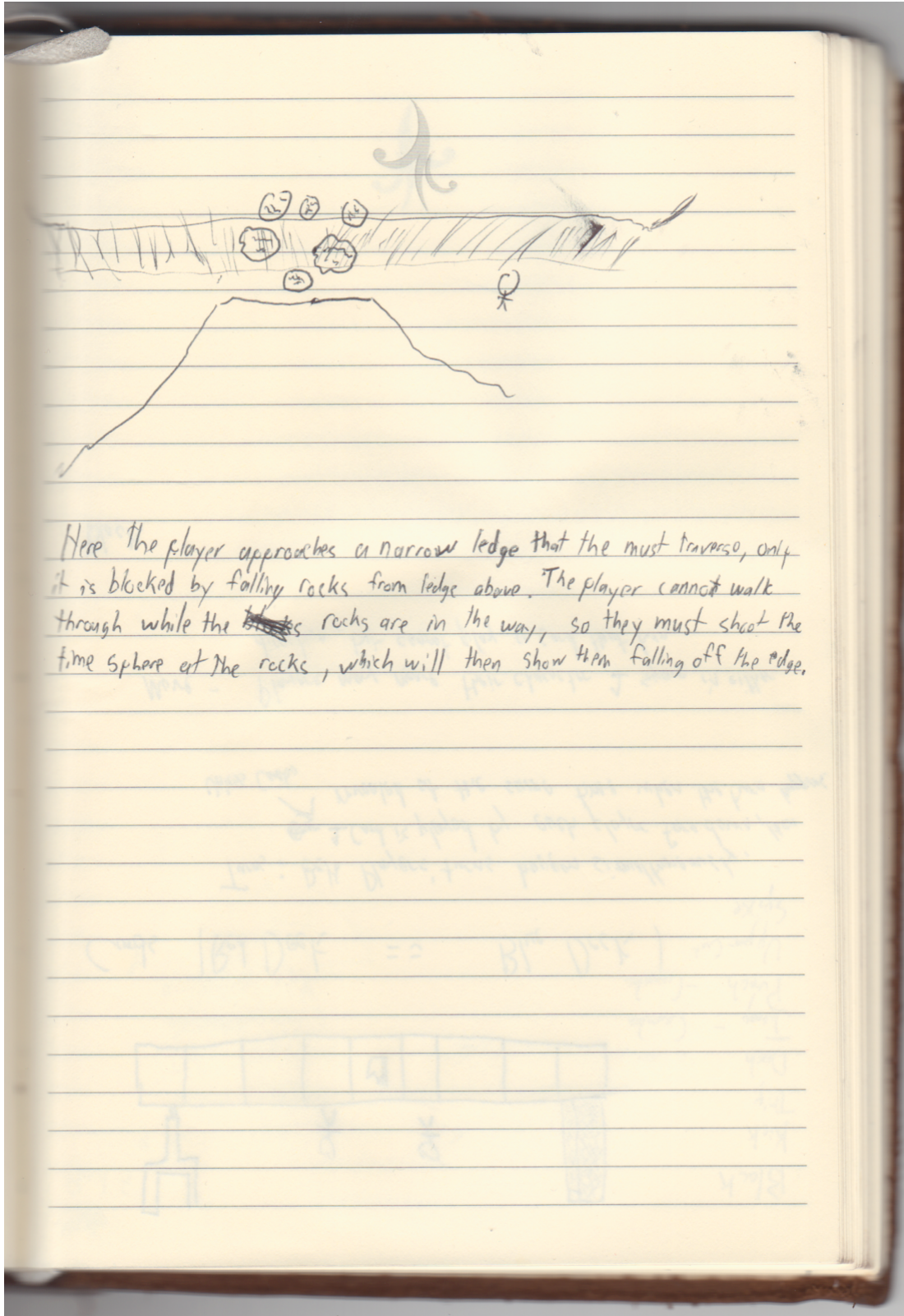
Mechanic Intro • Block Push

Pushing a box as a mechanic is rather boring. Because of that, we need the effects of solving a block puzzle to be impactful.

This scene is of a ledge and a bridge over an abyss leading to a large door. On the cliff, we present the player with a "switch" with block already on it, and a block off of a switch on the mirror side. This is a very simple way to introduce the idea of ~~pushing~~ getting the block onto the switch. If the player moves into the block, we need some way of clearly displaying that you are pushing the block. Possibly moving the camera back into a third person view when pushing a block. When the block gets on the "switch", I imagine it rising up into the air, the block glowing (runes on the side and carvings), and some kind of particle effect traveling to the destination of the solved switch.



An idea for an opening puzzle.



Here the player approaches a narrow ledge that they must traverse, only it is blocked by falling rocks from ledge above. The player cannot walk through while the ~~rocks~~ rocks are in the way, so they must shoot the time sphere at the rocks, which will then show them falling off the edge.

An area to demonstrate how the Chronosphere works.